Universidad de Granada

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Programa de Doctorado en Ciencias de la Computación
y Tecnología Informática

Sistemas de Clasificación Basados en Reglas Difusas para Problemas no Balanceados. Aproximaciones y Uso de Nuevas Estrategias para Resolver Problemas Intrínsecos a los Datos no Balanceados

Tesis Doctoral

Victoria López Morales

Granada, marzo de 2014

Editor: Editorial de la Universidad de Granada
Autor: Victoria López Morales
D.L.: GR 1901-2014
ISBN: 978-84-9083-079-6

Universidad de Granada

Sistemas de Clasificación Basados en Reglas Difusas para Problemas no Balanceados. Aproximaciones y Uso de Nuevas Estrategias para Resolver Problemas Intrínsecos a los Datos no Balanceados

MEMORIA QUE PRESENTA

Victoria López Morales

PARA OPTAR AL GRADO DE DOCTOR EN INFORMÁTICA

Marzo de 2014

DIRECTORES

Francisco Herrera Triguero y Alberto Fernández Hilario

Departamento de Ciencias de la Computación
e Inteligencia Artificial

La memoria titulada "*Sistemas de Clasificación Basados en Reglas Difusas para Problemas no Balanceados. Aproximaciones y Uso de Nuevas Estrategias para Resolver Problemas Intrínsecos a los Datos no Balanceados*", que presenta D^{a}. Victoria López Morales para optar al grado de doctor, ha sido realizada dentro del Programa Oficial de Doctorado en "*Ciencias de la Computación y Tecnología Informática*", en el Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada bajo la dirección de los doctores D. Francisco Herrera Triguero y D. Alberto Fernández Hilario.

El doctorando y los directores de la tesis garantizamos, al firmar esta tesis doctoral, que el trabajo ha sido realizado por el doctorando bajo la dirección de los directores de la tesis, y hasta donde nuestro conocimiento alcanza, en la realización del trabajo se han respetado los derechos de otros autores a ser citados cuando se han utilizado sus resultados o publicaciones.

Granada, marzo de 2014

El Doctorando

Los directores

Fdo: Victoria López Morales

Fdo: Francisco Herrera Triguero

Fdo: Alberto Fernández Hilario

Esta tesis doctoral ha sido parcialmente subvencionada por el Ministerio de Ciencia e Innovación bajo el Proyectos Nacional TIN2011-28488. También ha sido subvencionada bajo el programa de becas de Formación de Profesorado Universitario del Ministerio de Educación, en su Resolución del 11 de Octubre de 2010, bajo la referencia AP2009-4889.

Agradecimientos

Como la gratitud en silencio no sirve a nadie, quisiera aprovechar la oportunidad que me brindan estas líneas para acordarme de las personas que han ido poniendo su granito de arena para ayudarme a superar el reto que supone completar el desarrollo de una tesis doctoral.

En primer lugar, quisiera agradecer a mis directores de tesis Francisco Herrera y Alberto Fernández todo el tiempo y esfuerzo que han dedicado para introducirme en el mundo de la investigación. Sin su apoyo decidido, esta tesis no hubiera llegado a ser lo que hoy es. Su guía y consejo han demostrado ser un aliado valioso para ir avanzando en este recorrido.

Asimismo quisiera acordarme de todos aquellos que me han acompañado en el día a día de la investigación: de aquellos junto a los que comencé la tesis, Isaac, José Antonio, Álvaro, y de aquellos que nos ayudan de alguna manera con ella, Salva y Julián. También agradezco la compañía de los doctores más jóvenes del grupo, Nacho, Manolo Cobo, José García, Christoph, Fran y Michela o de los jóvenes doctores "de fuera", Mikel en Pamplona y Cristóbal en Jaén. También se agradece los consejos de la experiencia de Jesús y Rafa Alcalá, José Manuel Benítez o Chris Cornelis.

No puedo olvidarme de los doctorandos más noveles a los que les queda todavía un poquito más de camino por andar: Dani, Sara, Pablo, Sergio, Juanan, Raquel, Rosa y Lala, siempre con su optimismo y alegría. Finalmente, también incluir en este grupo a los ex-residentes de Orquídeas con los que comparto muchas mañanas un *fuzzy coffee*, Olmo, Rafa, Edu, Alberto e Irene.

I would also like to express my gratitude in these lines to Vasile Palade, the supervisor for my research visit at University of Oxford. Our talks about imbalanced datasets have been very valuable to understand some features of the problem and to redirect my focus from uncertain objectives towards more sensible paths.

En el plano personal, quisiera acordarme de mis padres José y M^{a}Victoria porque gracias a su apoyo y consejos, he podido día a día, cruzar el camino de la superación y abordar este desafío. Vuestra confianza y paciencia, los momentos de nervios y de tensión que habéis compartido conmigo me han servido de empuje para seguir adelante. Debo mencionar asimismo a mi tía Encarnación que también me ha acompañado en este camino de aprendizaje y evolución.

No menos importante ha sido el aliento de mis hermanos, Manuel e Isabel. Sabiendo que jamás encontraré la forma de agradecer su constante apoyo y confianza, sólo espero que comprendan que su presencia en todo momento ha sido uno de los mejores alicientes en seguir avanzando hacia la meta.

Finalmente, dicen que los últimos serán los primeros, quiero darle las gracias a Joaquín por su infinita paciencia, cariño y comprensión. Su mente inquieta me ha permitido ver un camino de luz cuando parecía que infinitos obstáculos me cerraban el camino. ¡Gracias por ser como eres y estar a mi lado!

Table of Contents

Part I. PhD dissertation

1. Introduction

Classification and prediction tasks are taking place constantly in our daily life. We can find various examples in real life carried out by experts in very different fields, such as medical diagnosis, pattern recognition, product rating... From a general point of view, the concept of classification covers every context where a decision is made based on the available information. However, the realization of this task can carry out many problems like slowness to finish or the context difficulty. Thereby, the development of automatic systems makes easier the work as they enable the obtaining of more accurate predictions. These systems are interesting because the data analysis performed by them does not have the subjectivity attached to human beings and because the capacity of analysis of an automatic method (volume of data) is always going to be bigger than a person capacity.

The classification problem is defined on the context of data mining (DM) categorized as a supervised learning task [TSK06]. By this, we mean that the set of examples we have available are labeled with the class they belong to. From here on, we have to learn and build a model or decision function that is able to return the class belonging to a new example based on its attribute values. This system is known as a classifier.

When trying to solve a given stage of the classification application, experts and researchers must know the data structure that they are processing so they can achieve the maximum accuracy related to all the concepts included in the problem. For example, there are many work areas where the class distribution is not balanced. Since most of the standard learning approximations consider a balanced training group, this leads to the obtaining of a suboptimal classification model, namely, a good cover on the examples that belong to the majority class (also known as negative class), while the minority examples (known as positive class) are more difficult to be properly identified. This problem is known as classification with imbalanced data [HG09, SWK09].

We must emphasize the importance of this problem, as it is related to complications in real-world domains. In these cases, a high cost is implied when examples of the positive class are classified in a wrong way as the examples that belong to the positive class are the most interesting from the learning point of view. Some of these real-world applications are medical diagnosis and fraud detection, among others. These problems typically present a small number of examples from the positive class as examples from this class are usually associated with exceptional or significant cases, or because the acquisition of instances is costly.

In the area of classification in general, and in classification with imbalanced data on particular, Computational Intelligence techniques (CI) [Kon05, Pet07] have shown to be a very robust tool

to obtain models with a high degree of confidence. Although there is no complete agreement with respect to a definition on CI, there is a widely accepted vision about areas included under this paradigm, such as Artificial Neural Networks, Fuzzy Logic and Evolutionary Computation. Among the available techniques in this field, linguistic fuzzy rule-based classification systems (FRBCS) [INN04] are a popular tool because of the interpretability of their models based on linguistic variables, which are easier to understand to final users or experts while obtaining good results in the area of imbalanced classification [FGdJH08, FdJH09, FdJH10].

Returning to the specific problem of classification with imbalanced datasets, we must note that since the initial studies it has been shown that the loss of efficiency is due to non-uniform distribution on classes. However, recent research suggests that the problem in this scenario is the synergy between the imbalance and some inner characteristics of data. Among these characteristics we can find the overlapping between classes [GMS08, DT10], the presence of small-disjuncts [Wei05, Wei10], the treatment of the borderline samples [DKS09, NSW10], the problem of noisy instances [BF99, SKVHF14], and finally, the different distribution on partitions of training and test data, which is known as dataset shift [Shi00, MTH10].

However, the difficulties in the obtaining of good performance models in classification problems and DM is not only directed towards the uneven class distribution. A new concept called *Big Data* has spread quickly in this framework [ADA11, Mad12]. This new scenario is defined by those problems that cannot be addressed effectively and/or efficient through the standard computational resources currently available. We must highlight that big data does not necessarily imply large volumes of information, it's just that the existing methods that are used to address the problem are not able to provide a classification answer within our requirements.

Our interest in this memory mainly lies in the study of the problem of classification with imbalanced datasets from the perspective of the data intrinsic characteristics that this type of problems display. We intend to perform a detailed analysis of the existing solutions to the problem to fully understand their behavior and discern which are more appropriate from a general point of view. With the information provided by this study, we intend to develop new learning methods with FRBCSs that will address the data intrinsic characteristics that degrade the performance of classifiers with imbalanced data and hence improve the behavior of the standard methodology defined to this area of DM. At last, our intention is to extend the study of classification with imbalanced data to the big data field. In particular, our goal is to analyze the scalability of the basic solutions of FRBCSs raised on, and propose new parallelization techniques to address this problem effectively.

To perform this study, this PhD dissertation is divided into two parts. The first one is dedicated to the statement of the problem considered and the discussion of results obtained; while the second part corresponds to the publications associated with the study.

In Part I of this document we begin with a section dedicated to the preliminaries related to the problem (Section 2), introducing the related information about approaches and other problems. Next, we define the open problems in this framework (Section 3) that justify the realization of this thesis as well as the proposed objectives (Section 4). Then, we include Section 5, discussion of results, which provides a summary of the developed studies and the most important results obtained for the objectives considered in this manuscript. Later, Section 6 summarizes the results obtained herein and presents some conclusions about them to, at the end (Section 7), discuss some aspects of future work that are open in the present memory.

Finally, to develop the objectives, Part II of the memory is constituted of five publications distributed in three parts:

- A Study on the Data Intrinsic Characteristics in Classification Problems with Imbalanced Datasets and Analysis of the Behavior of the Techniques from the State-of-the-art.
- Addressing the Data Intrinsic Characteristics of Imbalanced Problems using FRBCSs and Machine Learning Techniques.
- A study on the Scalability of FRBCSs for Imbalanced Datasets in the Big Data Scenario.

Introducción

Las tareas de clasificación y predicción están continuamente presentes en la vida cotidiana. Podemos encontrar diversos ejemplos en la vida real realizadas por expertos en diferentes ámbitos, como por ejemplo en diagnóstico médico, reconocimiento de patrones, calificación de productos, y un largo etcétera. Desde un punto de vista general, el concepto de clasificación cubre cualquier contexto en el que se toma una decisión en base a la información disponible. Sin embargo, la realización de esta tarea puede conllevar distintos problemas como la lentitud al realizarla o la dificultad del contexto. De este modo, el desarrollo de sistemas autómaticos no sólo puede ayudar a facilitar la labor a realizar, sino que además puede permitir efectuar mejor las predicciones, debido a que el análisis de los datos carece de la subjetividad inherente a los seres humanos y porque la capacidad de análisis de un método automático siempre será mucho mayor (el volumen de datos con los que puede trabajar es mucho mayor) que la capacidad de una persona

El problema de clasificación se enmarca dentro del contexto de la Minería de Datos (MDD) en su vertiente supervisada [TSK06]. Con ello nos referimos a que el conjunto de ejemplos de los que disponemos para realizar el aprendizaje están etiquetados con la clase a la que pertenecen. A partir de este punto debemos aprender y construir un modelo o función de decisión capaz de devolver la clase correspondiente a un nuevo ejemplo en base a los atributos que lo caracterizan. Este sistema se denomina un clasificador.

Cuando se pretende resolver una aplicación dada en el escenario de la clasificación, los expertos e investigadores deben conocer la estructura de los datos que gestionan para de esto modo alcanzar la máxima precisión para todos los conceptos incluidos en el problema [DHS01]. Por ejemplo, hay muchas áreas de trabajo en los que la distribución de las clases no es equilibrada. Puesto que la mayoría de las aproximaciones de aprendizaje estándar consideran un conjunto de entrenamiento equilibrado (o balanceado), esto conlleva la obtención de un modelo de clasificación sub-óptimo, es decir, un modelo con una buena cobertura de los ejemplos mayoritarios (también conocida como clase negativa), mientras que los minoritarios (conocidos como clase positiva) son más difíciles de discriminar. Este hecho se conoce como la clasificación con conjuntos de datos no balanceados [HG09, SWK09].

Debemos enfatizar la importancia de este problema, ya que está relacionado con problemas en dominios del mundo real que implican un alto coste cuando los ejemplos de la clase positiva se clasifican de manera errónea. Algunos de estos escenarios son diagnosis médica, sistemas de detección de intrusiones y detección de fraudes, entre otros. Los ejemplos de la clase positiva suelen ser poco numerosos en estos problems ya que suelen estar asociados con casos excepcionales o significativos, o porque la adquisición de estas instancias es costosa.

En el área de clasificación en general, y de clasificación con datos no balanceados en particular, las técnicas de Inteligencia Computacional (IC) [Kon05, Pet07] han mostrado ser una herramienta

muy robusta para la obtención de modelos con un alto grado de acierto. Aunque no existe un acuerdo total con respecto a una definición de IC, hay una visión ampliamente aceptada sobre las áreas que se enmarcan en este paradigma, como son las Redes Neuronales Artificiales, Lógica Difusa, y Computación Evolutiva. Entre las técnicas disponibles en este campo, los Sistemas de Clasificación Basados en Reglas Difusas (SCRBDs) Lingüísticas [INN04] son una herramienta popular debido a la interpretabilidad de sus modelos asociados basados en variables lingüísticas, que son más fáciles de comprender para los usuarios finales o expertos, además de obtener muy buenos resultados en el campo de acción de la clasificación no balanceada [FGdJH08, FdJH09, FdJH10].

Retomando el problema específico de la clasificación con conjuntos no balanceados, debemos destacar que desde los estudios iniciales se ha mostrado que la pérdida de rendimiento se debe a la distribución no uniforme de las clases. Sin embargo, recientes investigaciones sugieren que el problema en este escenario es la sinergia entre el desbalanceo y algunas características intrínsecas de los datos. Entre estas características podemos encontrar el solapamiento entre las clases [GMS08, DT10], la presencia de *pequeños datos disjuntos* (en inglés *small disjuncts*) [Wei05, Wei10], el tratamiento de los ejemplos frontera o *borderline* [DKS09, NSW10], el problema de las instancias con ruido [BF99, SKVHF14], y finalmente la distinta distribución en las particiones de datos de entrenamiento y test, conocido como *dataset shift* [Shi00, MTH10].

Pero la problemática en la resolución de los problemas de clasificación y MDD no solo se encuadra en el hecho de los conjuntos de datos no balanceados. Un nuevo concepto denominado *Big Data* se ha extendido rápidamente en este marco de trabajo [ADA11, Mad12]. Este nuevo escenario se define por medio de aquellos problemas que no pueden ser abordados de manera efectiva y/o eficiente a través de los recursos computacionales estándar de que disponemos actualmente. Debemos remarcar que *big data* no implica necesariamente amplios volúmenes de información, sino básicamente que los métodos existentes no son capaces de proporcionar una respuesta adecuada en estas situaciones.

Nuestro interés en esta memoria reside principalmente en el estudio de los problemas de clasificación con conjuntos de datos no balanceados bajo la perspectiva de las características internas que presentan este tipo de problemas. Pretendemos realizar un análisis pormenorizado de las soluciones existentes para conocer su comportamiento y discernir cuales son las más apropiadas desde un punto de vista general, con el objetivo de desarrollar nuevos métodos de aprendizaje con SCBRDs que permitan abordar las características intrínsecas de los datos, y por tanto mejorar el comportamiento de las metodologías estándar definidas para este área de la MDD. Por último, nuestra intención es la de extender el estudio de la clasificación con datos no balanceados al campo de *big data*. En particular, nuestro objetivo será analizar la escalabilidad de las soluciones básicas planteadas sobre SCBRDs, y proponer nuevas técnicas de paralelización para abordar este problema de manera efectiva.

Para llevar a cabo este estudio, la presente memoria se divide en dos partes, la primera de ellas dedicada al planteamiento del problema y discusión de los resultados y la segunda correspondiente a las publicaciones asociadas al estudio.

En la Parte I de la memoria comenzamos con una sección dedicada al "Planteamiento del Problema" (Sección 2), introduciendo éste con detalle y describiendo las técnicas utilizadas para resolverlo. Asimismo, definimos los problemas abiertos en este marco de trabajo que justifican la realización de esta memoria (Sección 3) así como los objetivos propuestos (Sección 4). Posteriormente, incluimos una sección de "Discusión de Resultados", Section 5, que proporciona una información resumida de las propuestas y los resultados más interesantes obtenidos en las distintas partes en las que se divide el estudio. La sección de "Conclusiones" (Sección 6) resume los resultados obtenidos en esta memoria y presenta algunas conclusiones sobre éstos. Finalmente, se comentan en la Sección 7 algunos aspectos sobre trabajos futuros que quedan abiertos en la presente memoria.

Por último, para desarrollar los objetivos planteados, la Parte II de la memoria está constituida por cinco publicaciones distribuidas en tres partes:

- Estudio de las Características Intrínsecas de los Datos en Problemas de Clasificación con Conjuntos de Datos No Balanceados y Análisis del Comportamiento de las Técnicas del Estado del Arte.
- Desarrollo de Aproximaciones para Resolver las Características Intrínsecas de los Problemas No Balanceados mediante SCBRDs y Técnicas de Aprendizaje Automático.
- Estudio de la Escalabilidad de los SCBRDs para Conjuntos de Datos No Balanceados en un Escenario de *Big Data*.

2. Preliminaries

Continuous improvements on information generation and storage have enabled in different knowledge and business areas, an extensive data gathering. The recognition of patterns in data, which is common in humans, is greatly automated using *Knowledge Discovery in Databases (KDD)*. KDD was defined in 1996 [FPSS96] like "the nontrivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data". Currently, it enforces two main roles: it has become fundamental in scientific research due to its analysis and knowledge discovery capabilities from available data; and it gradually expands with success its knowledge from traditional applications like marketing or finances, to other domains like industry, energy, medicine, bioinformatics or web analytics among others, in which the amount of information and the need to retrieve useful knowledge with a direct benefit, are increased by almost the same amount.

KDD is composed by a set of interactive and iterative steps such as data preprocessing, a search for interesting patterns with a concrete representation and the interpretation for these patterns (Figure 1). Although KDD is the appropriate name when it comes to this procedure, the term *Data Mining (DM)* [TSK06] is frequently utilized to refer to the complete process. This term represents the knowledge extraction from computed data [Pyl99] being actually the main task of the whole system. Depending on the objective, in DM it is possible to differentiate between predictive and descriptive tasks. For the first ones, the target is finding a model which allows the prediction of future behavior, usually by means of supervised learning. Within this group of MD tasks, classification, regression and prediction of temporal series can be named. Regarding descriptive DM, the process tries to build a model that describes information about the data subjacent problem employing unsupervised learning, and includes association rules extraction, clustering and summarizing among others tasks for DM.

An area with strong similarities with DM, is *Machine Learning (ML)* [Alp04]. Machine learning is a branch of artificial intelligence that concerns design and development of algorithms that are capable of learning behavior, patterns or concepts based in empirical data analysis, like sensor data o databases (which is the closest case for ML). In short, it is a tool that allow us to extract knowledge from a set of examples that represent the problem that we need to undertake.

In this memory, we will concentrate in the context of supervised learning and more specifically, in classification. With this background, classification refers to the process -with the previous knowledge

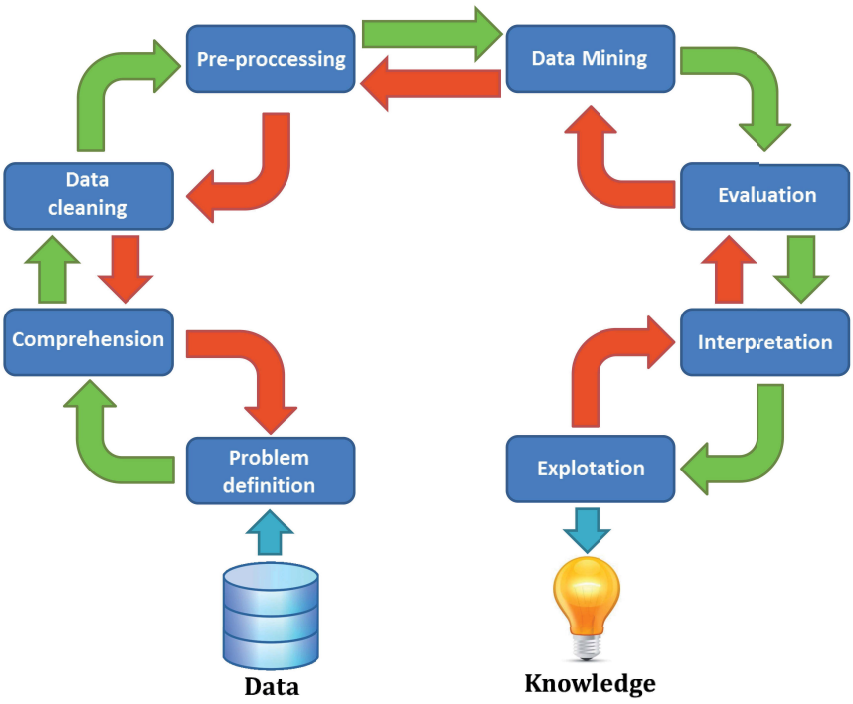


Figure 1: The KDD process

of certain classes or categories- where we establish a function or rule to pinpoint new predictions in some of the existing classes (supervised learning). A classifier receives as input a set of examples, labeled as training set, which learn the classification rule. Besides, during the validation process of a classifier, a set of examples is used, not known during the learning process named as test set and used to check the accuracy of the classifier. The classes are from a prediction problem, where each class corresponds to the possible output of a function to predict from attributes that describe the elements of a dataset.

When working with real applications of classification, we can see that they have as a common characteristic: a very different distribution of examples inside their classes. This situation is known as the problem of imbalanced classes [CJK04, HG09, SWK09] and is considered as one of the challenges in DM [YW06]. Specifically, in the context of binary problems, a class is usually represented by very few examples (known as positive class), while the other is described by many instances (negative class). The minority class is usually the main objective from the learning point of view and, for this reason, the cost related to a poor classification of one example of this class is greater than on the majority class.

An additional factor that affects the development of potential programs for the induction of knowledge is the *massive generation of data* in which we currently find ourselves immersed. This scenario has occurred for three main reasons [Kra13]:

1. Hundreds of applications like mobile sensors, multi-media social services, and other devices that are compiling information continuously.

2. The storage capacity has increased so much that data are cheaper than ever, making attractive to the customer to buy more space than to choose what to delete.

3. Matching learning methods and recovering of information have achieved a significant impro-

vement in the last years, allowing the acquisition of a higher level of knowledge from the data.

Specifically, Terabytes of data are written every day resulting in a large **Volume**; real-time requirements clearly imply a high **Velocity**, we can find a great **Variety** of either structured, semi-structured or even unstructured data; and data must be cleaned prior to integration on the system to maintain the **Veracity** [GGM12]. Those properties of 4V defines what is known as the problem of **Big Data** [ADA11, Mad12], having achieved the status of main theme of interest between academic and industry areas.

In addition to the importance of scalability in construction of models, is the construction of a symbolic structure in order to be useful, not only from a functional point of view, but also from the perspective of interpretability, i.e:, to seek models understandable to humans. A concept related to the interpretability of models is the CI [Kon05] (also known as *Soft Computing*). This concept encompasses those models or techniques that try to seek inexact solutions to computer problems that are too complex so we cannot obtain an exact solution in a polynomial time. Logically, given the amount of data that we are working in DM, this idea encompasses most of the methodologies that can be applied. Among the most extended are: evolutionary computation [Gol89], fuzzy logic [Zad65], neural networks [Gur97], case-based reasoning [AKA91] or any hybridization on the above.

Within the context of CI, our framework for the development of the thesis is focused on the use linguistic FRBCS [INN04]. The main reason is due to the advantage associated with the obtaining of easy interpretable models, based on linguistic variables, which are simpler to understand to the final or expert user. Additionally, this type of systems have performed well when applied to the classification with imbalanced datasets.

The following subsections detail each of these aspects that are directly related herein. In Section 2.1, we introduce in detail the problem of classification with imbalanced datasets. Later, in Section 2.2, we define the area of work concerning the concept big data. Finally, in Section 2.3, we describe the characteristics of linguistic FRBCS.

2.1. Classification problems with imbalanced classes

Within the real problems of matching learning in general, and classification in particular, researchers find that the example distribution in different classes or concepts that represent the dataset is not uniform. This problem is observable in many examples, such as fraud detection, risks management, texts classification, medical diagnosis, and many other domains in which this characteristic is implicitly attached to the problem, because fortunately, there are usually very few anomalous cases in comparison with normal cases. Another situation which can lead to the appearance of this type of sets occurs when the data recollection process is limited (due to economical or private reasons). It is important to note that this type of datasets with *imbalanced classes* differ from standard datasets not only in the imbalance between classes, but also into the growing importance of the minority class, traditionally identified each as *positive class*.

Despite showing a fairly common occurrence and a strong impact on day life applications, the problem of imbalanced classes has not been properly solved by matching learning algorithms, since they assume balanced class distributions or equal classification costs for all classes.

In fact, most of the learning algorithms aim to obtain a model with a high accuracy on prediction and a good generalization ability. Nevertheless, those algorithms that perform well in the context of standard classification not necessarily achieve the best performance for imbalanced datasets

[FGL+10]. We note therefore that the bias on classification algorithms for examples of the majority class [SWK09, HG09] is the most direct consequence derived from the unequal distribution of classes. When the search process is guided by the standard accuracy measure, it benefits the covering of the majority of the examples. Secondly, the classification rules predicting the positive class are often highly specialized so their coverage is very low, and therefore, they are discarded in favor of more general rules, for example, those that predict the negative class.

In practical applications, the rate of the minority over the majority class may be drastic when we have 1 example versus 10, 1 versus 100 or 1,000. In our work, we have the imbalance ratio or IR [OPBM09], defined as the fraction between the number of examples of the majority class and the minority class, to organize the different sets of data according to the value of IR.

Unfortunately, the problem of imbalanced classes usually appears in combination with different data intrinsic characteristics. This imposes additional constraints during the learning stage. First, we highlight the presence of areas with a high overlapping between classes, which effect is much more negative as when we want to discriminate the examples of the positive class [GMS08, DT10]. Additionally there may also be small groups of examples (*small-disjuncts*) of the minority class that can be treated mistakenly as noise, and therefore ignored by the classifier [OPBMG+09, Wei10]. The existence of even a few noisy examples can degrade the identification of the minority class, because it has a lower number of examples [SKVHF14]. Finally, we should note the case of *dataset shift*, based on the different distribution of data partitions between training and test [MTH10].

It appears therefore a high difficulty to achieve the final goal of developing a classifier that reaches a high precision, on both the positive and negative classes of the problem. That is why the area of imbalanced classification datasets has been widely studied through last years [HG09, SWK09]. A large number of solutions has been developed for this task, and can be categorized into three groups:

- Sampling data: in which training instances are modified to achieve a distribution of class classes more balanced in order to enable the classifiers to work in a similar way as the standard classification [BPM04].
- Algorithmic modification: this procedure is oriented towards the adaptation of learning models, so we can tune them to the properly addressed the uneven class distribution [LTY13, ZHC13].
- Cost-sensitive learning: such solutions incorporate approximations on the level of data, on algorithmic level, or even on both levels together. Higher costs are considered due to bad classification of examples of the positive class compared to the negative class and, therefore, tries to minimize the level of associated cost to the overall problem [BP10, ZLA03].

In addition to the previous techniques, recently, ensembles of classifiers have appeared as a possible solution on the problem of class imbalance, awaking a great interest among researchers [KR14, LWZ09, SKVHN10, SKWW07, VHKN09, WY13]. The *ensemble* based methods are modified or adapted by combination among the ensemble learning algorithm itself and either technique described above, to namely, either as data level or by algorithmic modifications based on cost sensitive learning.

In the case of adding a data level approach for learning algorithm ensemble, the new hybrid method usually preprocesses the data before the formation of each classifier. In addition, in cost sensitive ensembles type, instead of the modifying the base classifier towards the end of accepting costs in the learning process, what they do is guide the minimization of costs through ensemble

learning algorithm. Thus, we avoid the modification on the based learning method, but the main drawback, which is the definition of costs, will be present.

2.2. Data Mining and Big Data

It is very challenging to present a correct definition of the term *Big Data* [Kra13]. This term was coined very recently, when data intensive companies started to face large collections of data, at a petabyte scale. In fact, it is estimated that a 90 % of the data currently available has been created within the last two years [WZWD14]. The sources of this huge amount of information are very sparse: Applications tracking clicks in websites, transaction records, sensors, social networks, scientific applications ...

Initially, we might argue that the term *big data* is only related with the size of the data. But the truth is that this **Volume** of data is not the only property inherent to the big data realm. Besides **Volume**, it is very easy to realize that large collections of data will most likely show a high degree of variability, heterogeneous structures, and a remarkable **Variety** regarding the way in which information is represented. For example, different software implementations of data management systems will involve the use of different protocols and data schemes [SJ12]. Also, the data format here plays a fundamental role when determining how it will be processed (as data management systems will not deal with images in the same way as they do with, for example, text files).

Velocity is another fundamental property of the topic at hand. Nowadays, users demand for an *acceptable response time* when working with data processing applications. Obviously, this factor will be mostly affected by the computational resources available (as we cannot compare a personal computer with a data processing center of a large company in terms of processing power).

Finally, big data applications must also maintain the **Veracity** of information; that is, diminish the effect of anomalies and noise within the data.

These factors are commonly known as the four V's of big data, and form the basis of most of the current definitions of the term, such as Gartner's *"Big data is high volume, high velocity, and/or high variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization"*.

However, big data challenges are mainly motivated by two issues [LJ12]:

- The storage and management of large volumes of data. This problem is closely related with traditional entity-relation database management systems. Commercial solutions often offer good scalability, being able to manage petabytes-sized databases. However, besides their high cost - regarding both money and computational resources - they also are very restrictive when it comes to import data from its original representation. Open source systems, such as MySQL, are less prone to show this problem, but they often show a much more limited scalability.

- The exploration and analysis of the data, aiming to discover useful knowledge for future applications [WZWD14]. Standard analytics are usually based upon entity-relation schemes, and developed through various SQL queries. However, besides the difficulties managing and storing data, the problem here is the lack of statistical support to go beyond mere aggregations of data. And even if database applications would be able to provide such support, they still could not provide it in an efficient way, considering the large amount of data that they must manage.

Distributed [RJBF+80] and parallel [DGS+90] databases could be used to address the first issue, enabling existing systems to deal with a high workload of analytics-related tasks. However, they again face very serious problems when big data comes to the scene, as they require very high hardware requirements. Also, current applications need to manage unstructured or semi-structured data, which becomes an additional challenge for this kind of systems.

An alternative has been proposed to the traditional databases, according to these facts: A new technology for data management, known as *Not Only SQL* (NoSQL) [HHLD11, CDG+08], which basically consists on storing the information as *Key-Value* keys, providing horizontally distributed scalability. It is important to remark that NoSQL databases provide with a flexible data model, supporting different data representations; thus, big data applications are quickly adopting NoSQL as their main option for storage.

A second point of view is focused on the programming models that are adopted to analyze the data, most of which are commonly based on parallel computing [SAM96], such as, for example, the *Message Passing Interface* (MPI) model [GLDS96]. The challenges here are to provide a proper way to access to the data and to ease the development of specific software according to the requirements and limitations of the common programming paradigms.

For example, standard DM algorithms require all data to be loaded in the physical memory. This is a challenging problem in big data, because most of the times data is stored throughout different machines/networks, and thus gathering it requires a large amount of network-based communication and input/output operations. And even if this would be feasible, there is still the necessity of providing an extremely large amount of physical memory to store all the data needed to run the computing programs.

A new generation of systems has been developed in order to provide a proper way of tackling the aforementioned issues, with MapReduce [DG08] and Hadoop [The12, Lam11] - its open source implementation - as its most representative members both in industry and academia.

This new paradigm avoids the above limitations regarding the necessities of loading the data, storing it in physical memory, or even the use of SQL. Instead, developers now can code their programs using this new model, which enables them to parallelize the applications automatically. This is achieved by the definition of two simple functions - well-known in the functional programming paradigm - denoted as *Map* and *Reduce*. *Map* can be used to group and split data, whereas *Reduce* aim is to perform the necessary computations to produce the final output of the program.

Both functions work by dividing the input dataset into independent subsets, which can be processed in parallel by *Map* tasks. Then, Hadoop sorts the outputs of the *Map* tasks and convert them to inputs for the *Reduce* tasks. In more detail, it works as follows [WYLD10]:

- **Key/Value pairs** are the processing primitives. The *Map* functions are applied to every input key/value pair, generating an arbitrary number of intermediate key/value pairs.
- These intermediate values are provided to the *Reduce* function, by using an iterator able to manage very large lists of pairs (often too large to be stored in the physical memory). The *Reduce* functions are then applied to all the values associated with the same intermediate keys, generating an arbitrary number of output key/value pairs.
- As an optimizing step, MapReduce introduces the use of *Combiners*, which are able to work directly with the output of the *Map* functions. This allows to save a huge amount of network traffic, since it does not require the intermediate step of sorting the keys before feeding them into the *Reduce* tasks.

- The final component of MapReduce is the *Partitioner*, which is in charge of splitting the intermediate keys and assigning the key/value pairs to the *Reduce* tasks. The default *Partitioner* computes a hash value of the key, and computes the modulus of dividing it by the number of *Reduce* tasks, using it as an index to deliver approximately the same number of keys to each task.

We must highlight that, in the four points previously arisen, the last two functions are optional during the MapReduce process and its usage is limited to those jobs that need to be intensely optimized. In a general case, Hadoop-based programs (Figure 2) are managed by *Map* function calls, which are distributed throughout multiple machines by partitioning automatically the input data into M slots (so they can be processed in parallel by different machines); and *Reduce* function calls which are distributed by partitioning the key space into R chunks, with R specified by the user.

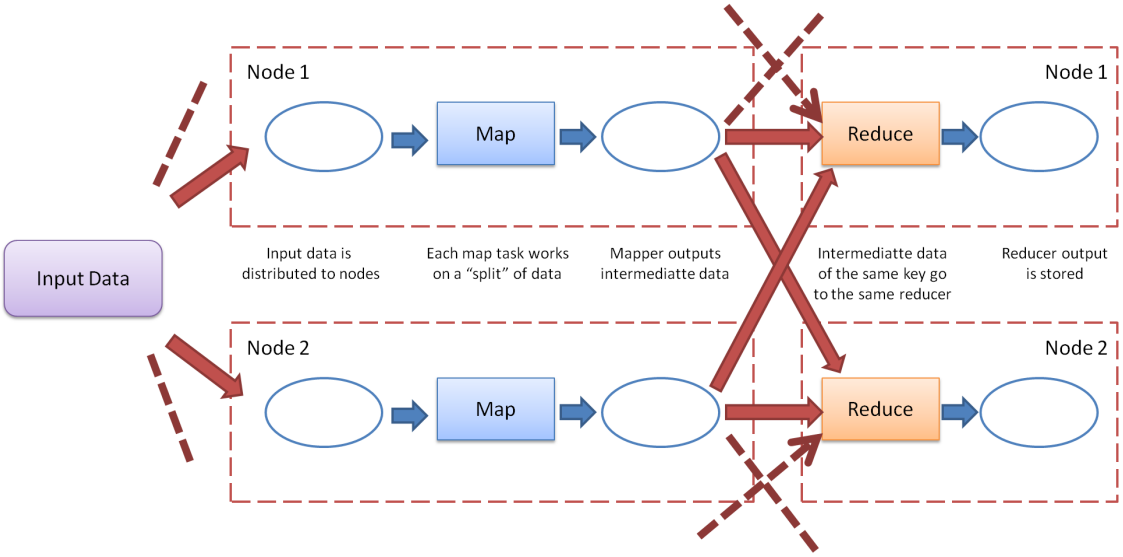


Figure 2: Complete flowchart of an operation in MapReduce

In summary, Hadoop-based systems are oriented towards the distribution of datasets in a cluster (which does not necessarily has to be formed by high performance machines) to parallelize computations in the nodes. The rationale here is that mapping functions can be defined to create intermediate <key, value> tuples and reducing functions can be used to process the data spatially, avoiding the rather costly alternative of gathering the data in a core machine. In this way, a representative example could be to count the number of occurrences of every word in a large collection of documents. Here, Hadoop will proceed to use mapping functions to broadcast every word with the count of the times that it appear in every single document. Then, reducing functions will sums those values along each distinct word, obtaining as a result the final count.

2.3. Classification Systems based in linguistic fuzzy rules

Fuzzy systems are one of the most important areas where the Fuzzy set theory is applied. In the environment classification, a model structure is used in the form of FRBCSs. The FRBCSs constitute an extension of rule-based systems, since they use type rules like *IF-THEN*, whose antecedent (and in some cases consequent) are composed of fuzzy logic statements, instead of

conditionals with a traditional format. Additionally, they have demonstrated their ability to so solve classification problems or DM in a large number of applications [Kun00, INN04].

The most common type of FRBCSs are *linguistic* FRBCSs or *Mandani* type [Mam74], which they have the following format:

R_i : IF X_{i1} IS A_{i1} AND $\cdots$ AND X_{in} IS A_{in} THEN C_k WITH PR_{ik}

where $i = 1$ to M, and being X_{i1} to X_{in} input variables and C_k the output class associated to the rule, being A_{i1} to A_{in} antecedent labels, and PR_{ik} the weight of the rule [IY05] (usually the certainty factor associated with the class).

All FRBCSs are composed of two basic components such as knowledge base (KB) and the module with the inference system. The KB is formed by two components, a Data Base (BD) and a Rule Base (BR):

- The DB contains the linguistic terms considered in linguistic rules and membership functions that define semantics of fuzzy labels. Thus, each linguistic variable included in the problem would have associated a fuzzy partition whose elements are linked with each linguistic term. Figure 3 shows an example of a fuzzy partition with five labels.

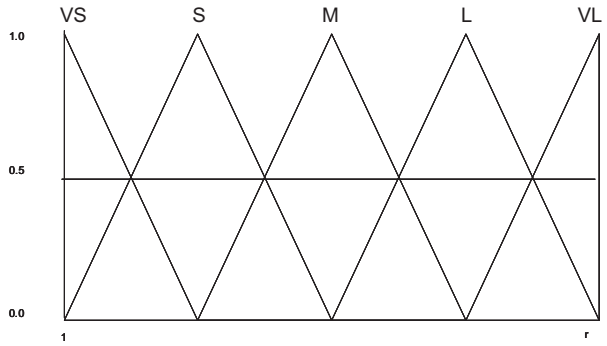


Figure 3: Fuzzy partition example

  This can be considered as an approximation to discretization for continuous domains on where we establish a degree of membership of the items (labels), where we include an overlap between them, and the inference engine operates pairing between patterns and rules, providing an output according to the rule consequents with a positive match. The determination of the fuzzy partitions is crucial in fuzzy modeling [ACW06], and the granularity of the fuzzy partitions plays an important role on the behavior of FRBCSs [CHV00].

- The RB, formed by a set of linguistic rules that are directly grouped together by aggregating them with an equal importance level. In other words, you can fire multiple rules at the same time with the same input.

The module with the inference engine includes:

- A *fuzzification interface*, which has the effect of transforming *crisp* data in fuzzy sets.
- An *inference system*, which taking received data from the fuzzification interface, it uses the information contained on the KB to do an inference using a fuzzy reasoning method (FRM).

  Specifically, if we consider a new pattern on $X_p = (X_{p1}, \ldots, X_{pn})$ and a RB formed by L fuzzy rules, the inference engine steps for classification are as follows [CdJH99]:

1. *Matching Degree.* It calculates the *strength of activation of the IF part using for all the rules in the RB with the X_p pattern*, using a conjunction operator (usually a T-norm).

$$\mu_{A_j}(X_p) = T(\mu_{A_{j1}}(X_{p1}), \ldots, \mu_{A_{jn}}(X_{pn})), \qquad j = 1, \ldots, L. \tag{I.1}$$

2. *Association degree.* We calculate the *association degree of the X_p pattern with the M classes according to each rule in RB.* When considering rules with only a consequent (like the ones presented in this section) this association degree only refers to consequent class of the rule ($k = C_j$).

$$b_j^k = h(\mu_{A_j}(X_p), RW_j^k), \qquad k = 1, \ldots, M, \quad j = 1, \ldots, L. \tag{I.2}$$

3. *Degree of consistency of the classification pattern for all classes.* We use an aggregation function that combines the positive degrees of association calculated on the previous step.

$$Y_k = f(b_j^k,\ j = 1, \ldots, L \text{ y } b_j^k > 0), \qquad k = 1, \ldots, M. \tag{I.3}$$

4. *Classification.* We apply a decision function F about the consistency degree of the system for the pattern classification in all classes. This function will determine the l class label corresponding to the maximum value.

$$F(Y_1, \ldots, Y_M) = l \quad \text{so that} \quad Y_l = \{max(Y_k), k = 1, \ldots, M\}. \tag{I.4}$$

Finally, the generic structure of a FRBCS is shown on Figure 4.

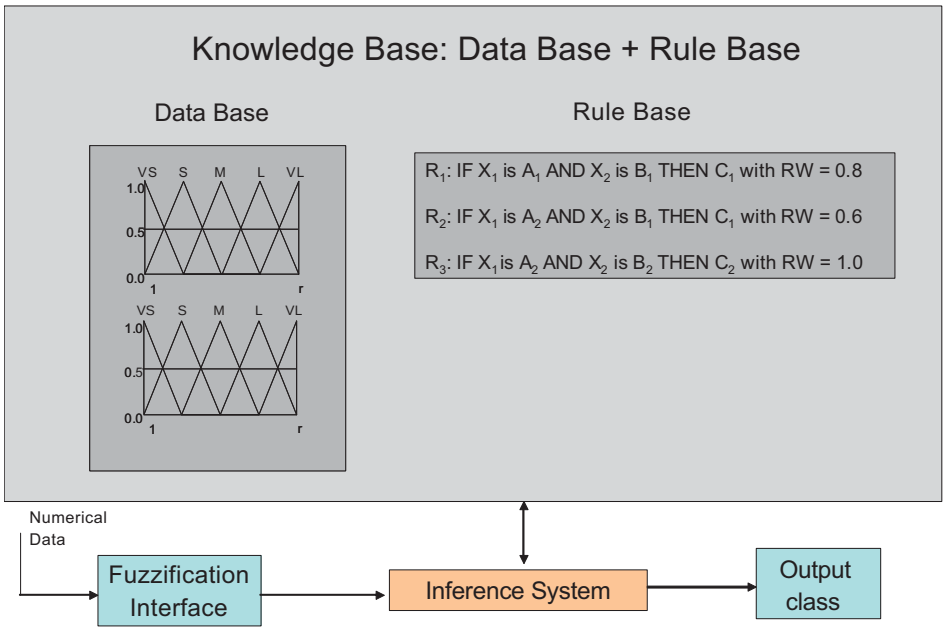


Figure 4: FRBCS structure

3. Justification

After the presentation of all the main concepts related to the topic, we identified some open problems that were interesting to be further analyzed:

- In the scenario of classification with imbalanced datasets, there are some works that review the associated issues to this problem [HG09, SWK09]. These contributions aggregate some of the solutions that have been given to the problem and they discuss some related aspects like assessment metrics and the relationship between real-world problems and imbalance. However, these texts do not perform an experimental comparison among the diverse proposals available in the state-of-the-art. Furthermore, the different type of solutions that are given to the problem are grouped by families which are categorized with respect to some specific trait that differentiates them. There is not a comparison detecting the behavior of methods belonging to different families of methods which could be helpful to select an appropriate alternative among all the available approaches.

- Furthermore, the existing studies on classification with imbalanced datasets are mainly focused on dealing with the uneven class distribution and trying to find a balance between generalization and proper identification of the underrepresented class. These surveys try to explore the nature of the problem, however, they do not analyze in depth some data intrinsic characteristics that may have an excessive negative effect over the classification of these datasets. Moreover, some of these characteristics have been sketchily considered without establishing a baseline to compare their impact over imbalanced datasets.

- Among the data intrinsic characteristics that degrade the performance of classifiers in the imbalanced scenario, we can identify the presence of small disjuncts, the areas of overlapping between the classes or the presence of borderline and/or noisy examples. Fuzzy rule-based classification systems have demonstrated their good performance in the imbalanced scenario [FGdJH08, FdJH09] providing an effective tool to achieve good classification results while providing an interpretable model to the end user. Furthermore, FRBCSs have demonstrated their robustness in the presence of noise [SLH10]. In this manner, it is interesting to design a new FRBCS that is able to be adapted to different data areas to address skewed class distributions together with some of the data intrinsic characteristics that deteriorate the classification performance.

- Another data problem that affects the classification with imbalanced data is the dataset shift problem. The issue of dataset shift often appears on real world data mining applications, mostly due to sample section biases when obtaining the training data. The relationship between the class imbalance problem and dataset shift has been hinted [MTH10], however, this issue has been previously studied from a data level point of view and without comparing the impact in the classification performance over some well-known machine learning methods.

- The enormous increment of data generation and storage that has taken place in the last years has become a challenge to standard machine learning techniques. In this situation, the knowledge extraction process is desired to be able to manage and include this new information to the learning step in a reasonable amount of time. Unfortunately, one of the simplest the more popular approaches to deal with this situation are based on a parallel divide-and-conquer strategy, where the available data is distributed among several processing nodes. This way of

working has a pernicious effect on the performance of classifiers in the imbalanced scenario as this division promotes the small sample size problem and the generation of small disjuncts. Furthermore, as it is a topic that has emerged in the late years, there are no works that analyze how to tackle imbalanced big data problems.

4. Objectives

The aim of this thesis is to perform an in-depth study of classification with imbalanced datasets focusing on the performance of available methods and to analyze the issues that degrade the performance in this scenario, with an especial focus to the usefulness of fuzzy rule-based classification systems to address this type of problems. This thesis is organized in several objectives which gather the open problems that were described in the previous section and which summarize the main goal:

- *To determine the behavior of the available techniques for classification with imbalanced datasets.* Considering the numerous methods available for classification with imbalanced datasets, we aim to perform an study that is experimentally able to determine the performance of the different groups of families of methods that are able to deal with these datasets, namely, preprocessing methods, cost-sensitive learning and ensemble based classifiers. In order to do so, we include methods from different learning paradigms such as decision trees, instance-based learning, support vector machines and fuzzy rule-based classification systems. Moreover, we do not only want to know how these families of methods work among themselves, but also how they behave when they are contrasted with other methods that belong to a different family.

- *To perform a thorough analysis on the data intrinsic characteristics that difficult the learning in the presence of imbalanced datasets.* We want to evaluate the impact of the data intrinsic characteristics that have been said to strongly influence the performance of classifiers when dealing with imbalanced datasets. We think that it is interesting to bring together all the data problems that have been brought up by other authors. Furthermore, it is also interesting to perform an experimental analysis that compares the influence and the degradation that these data intrinsic characteristics inflict over the classifiers and the correct identification of samples that belong to each class.

- *To improve the effectiveness in the classification of imbalanced datasets considering the data intrinsic characteristics using fuzzy rule-based classification systems.* Among the methods available for classification, fuzzy rule-based classification systems have been considered effective tools for classification as they provide a good trade-off between the precision achieved by the model and the accuracy obtained. This type of methods have demonstrated its good performance with imbalanced datasets [FGdJH08, FdJH09] and they enable the obtaining of new methodologies that are able to consider the data intrinsic characteristics previously studied to improve the effectiveness in classification in this scenario. The nature of fuzzy methods is able to improve the performance when the noise is involved. Furthermore, the usage of a hierarchical method allows the management of different granularity levels. These different granularity levels are able to better divide the regions with overlapping between the classes, to better differentiate the borderline instances that belong to each class and to reduce the number of small disjuncts that are created when the fuzzy rules are generated.

- *To examine the impact of dataset shift as a data intrinsic characteristic when imbalanced datasets are considered.* Dataset shift is another of the data intrinsic characteristics that has an impact on the performance that classifiers may obtain when confronted with an uneven class distribution. This dataset shift often appears on real world data mining applications, however, it can also be introduced when a cross validation procedure is used. In this manner, it seems sensible to study how several classifiers that come from different machine learning approaches behave when they are applied in a situation where dataset shift is alleviated in contrast with a situation where dataset shift is more tangible.
- *To evaluate the suitability of fuzzy rule-based classification systems for imbalanced big data problems.* As real-world problems usually present a skewed class distribution, it is natural to assume that in the big data scenario, where massive amounts of data are collected trying to represent reality as close as possible, this distribution is also noticeable. However, there are not works that have evaluated nor addressed imbalanced big data. Furthermore, big data introduces a certain degree of uncertainty and ambiguity as the data collected comes from different sources, is incomplete and sometimes it cannot be trusted. Therefore, FRBCSs seem to provide a suitable solution to this type of problem as they are inherently able to deal with this type of information. It is necessary to check if the current FRBCSs algorithms are able to directly provide an answer in this situation or if it is needed to somehow modify the current approaches and adapt them so that they can provide a suitable resolution to imbalanced big data in a reasonable response time.

5. Discussion of results

In this section, a brief summary of the different proposals that have been included in this dissertation are presented, describing their main contents, a brief discussion about the obtained results and the associated journal publications.

5.1. A Study on the Data Intrinsic Characteristics in Classification Problems with Imbalanced Datasets and Analysis of the Behavior of the Techniques from the State-of-the-art

The problem of classification with imbalanced datasets has attracted the attention of researchers in the last decade as it is present in many real-world applications. Numerous proposals to deal with imbalanced datasets have been presented to help to overcome the problem and obtain a correct identification of samples that belong to the minority class.

To gain a deep understanding about classification with imbalanced datasets and the issues that need to be addressed to improve the performance of methods that are able to address this problem, we need to perform an in-depth experimental study. To complete both objectives, we needed to thoroughly revise the state-of-the-art related to classification with imbalanced data. In doing so, we appreciated that even when numerous proposals had been given to approach the problem, they had not been experimentally compared difficulting the selection of a solution from a practitioner point of view.

In this manner, we decided to perform an extensive analysis of diverse solutions recommended for skewed class distributions. We started performing a comparison between preprocessing techniques

and cost-sensitive learning. To do so, we selected several algorithms from diverse classification paradigms, namely, decision trees, support vector machines, fuzzy rule-based classification systems and instance-based learning. The results were not able to find a superiority of one approach over another even when slight differences were found for certain baseline classifiers.

As this first study was not conclusive enough, we decided to extend the previous comparison adding some ensembles of classifiers to the survey, and we extended the number of preprocessing approaches and cost-sensitive learning techniques considered. Furthermore, instead of comparing all the methods alltogether, we decided to compare the diverse methodologies in families comparisons, and only the methods that showed a better performance were selected to evaluate their performance with respect to methods belonging to other families. In general, the proposals showed a more or less similar behavior, where the ensembles of classifiers obtained better results when the base classifier is a weak learner.

The study of the state-of-the-art has not only provided an insight about the approaches that can be used to tackle the problem of imbalanced classification but also it has provided information about what we have called the data intrinsic characteristics. The data intrinsic characteristics are some features that can be appear in the data and that negatively affect the performance of methods in imbalanced datasets. These characteristics can also emerge in balanced datasets, however, their influence in the performance of classifiers in the imbalanced scenario is much more disastrous than in the general case.

This data intrinsic characteristics include the presence of small disjuncts, the lack of density and information in the training data, the problem of overlapping between the classes, the impact of noisy data in imbalanced domains, the significance of the borderline instances to perform a correct identification of samples that belong to each class and the differences between the training and test data, also known as dataset shift. We have thoroughly discussed how they affect the classification performance in imbalanced data and we have included some experimental results that try to establish a baseline between the impact of each one of this data intrinsic characteristics.

The journal articles associated to this part are:

- V. López, A. Fernández, J. G. Moreno-Torres, F. Herrera, Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics. Expert Systems with Applications 39:7 (2012) 6585–6608, doi: 10.1016/j.eswa.2011.12.043
- V. López, A. Fernández, S. García, V. Palade, F. Herrera, An Insight into Classification with Imbalanced Data: Empirical Results and Current Trends on Using Data Intrinsic Characteristics. Information Sciences 250 (2013) 113–141, doi: 10.1016/j.ins.2013.07.007

5.2. Addressing the Data Intrinsic Characteristics of Imbalanced Problems using FRBCSs and Machine Learning Techniques

In the previous section, we introduced the data intrinsic characteristics that have an impact on the classification performance of the learners. This knowledge has enabled the identification of issues that need to be addressed to improve the performance of existing classifiers. Among the classifiers that provide a robust model in the presence of noise (one of the problems that negatively influence the presence of the imbalance), FRBCSs provide an interpretable model while maintaining a reasonable predictive capacity. Therefore, in Section 5.2.1 we present a proposal that describes a FRBCS that is designed to adapt its behavior considering the data intrinsic characteristics that

may affect the specific data case that is managed. Furthermore, some other intrinsic characteristics may also influence the classifiers, like the dataset shift. In this manner, we present a study in Section 5.2.2 that analyzes the performance of several approaches to machine learning over data that is less affected by dataset shift in contrast with data which is more influenced by the dataset shift problem.

5.2.1. A Hierarchical Genetic Fuzzy System Based On Genetic Programming for Addressing Classification with Highly Imbalanced and Borderline Data-sets

In this work, we propose the usage of a hierarchical environment to improve the performance of linguistic FRBCS, preserving the original descriptive power of fuzzy models and augmenting its precision improving the performance in areas of the data that are especially difficult to properly identify known as GP-COACH-H (Genetic Programming-based learning of COmpact and ACcurate fuzzy rule-based classification systems for High-dimensional problems Hierarchical).

The hierarchical environment that allows the usage of different granularity levels alleviates some of the data intrinsic characteristics that aggravate the performance of classifiers in the imbalanced scenario. The idea is to establish two types of rules, specific rules that posses a high granularity level, and more general rules with a low granularity level. In this manner, the number of generated small disjuncts is reduced, and therefore, their impact is alleviated. Furthermore, it is also able to address the overlapping between the classes, as this method increments its granularity when samples from both classes are mixed to some extent, and thus improving the identification of minority class instances in this situation. Moreover, this method is also able detect borderline examples, as it modifies its granularity level to properly identify and differentiate the class frontiers.

GP-COACH-H follows a genetic programming-based algorithm for the learning of fuzzy rule bases using a genetic cooperative-competitive learning approach that generates DNF fuzzy rules. It is based on the GP-COACH algorithm [BRdJH10] and follows a hierarchical fuzzy scheme similar to HFRBCS(Chi) [FdJH09].

This method is divided in three different steps. First, a preprocessing stage is applied using the SMOTE algorithm [CBHK02] to balance the class distribution. Then, a hierarchical data base is created over the balanced dataset. The generation of the hierarchical data base is done by the generation of triangular equally distributed membership functions that are built in two levels and the generation of the hierarchical rule base is performed by a genetic programming procedure that builds rules with two granularity levels that try to cover as many samples as possible while being simple and compact. Finally, a step to refine the hierarchical knowledge base is applied. Figure 5 depicts a flowchart of the GP-COACH-H algorithm.

To demonstrate the effectiveness of the proposal we consider forty-four highly imbalanced datasets (datasets with an imbalance ratio higher than nine) in our experimental study and we compare the results with the baseline algorithms, namely, the original GP-COACH algorithm over a dataset preprocessed with SMOTE and the previous hierarchical proposal HFRBCS(Chi) that served as inspiration for GP-COACH-H. The comparisons performed demonstrate the necessity of using the preprocessing step for highly imbalanced datasets. Furthermore, GP-COACH-H displays a good performance in this scenario, where the data intrinsic characteristics seem to deteriorate the classifiers performance. This good behavior is supported by the corresponding non-parametric statistical tests.

On the other hand, we have also tested the model over thirty borderline datasets which introduce different disturbance levels that allow the study of the performance over samples that are clearly

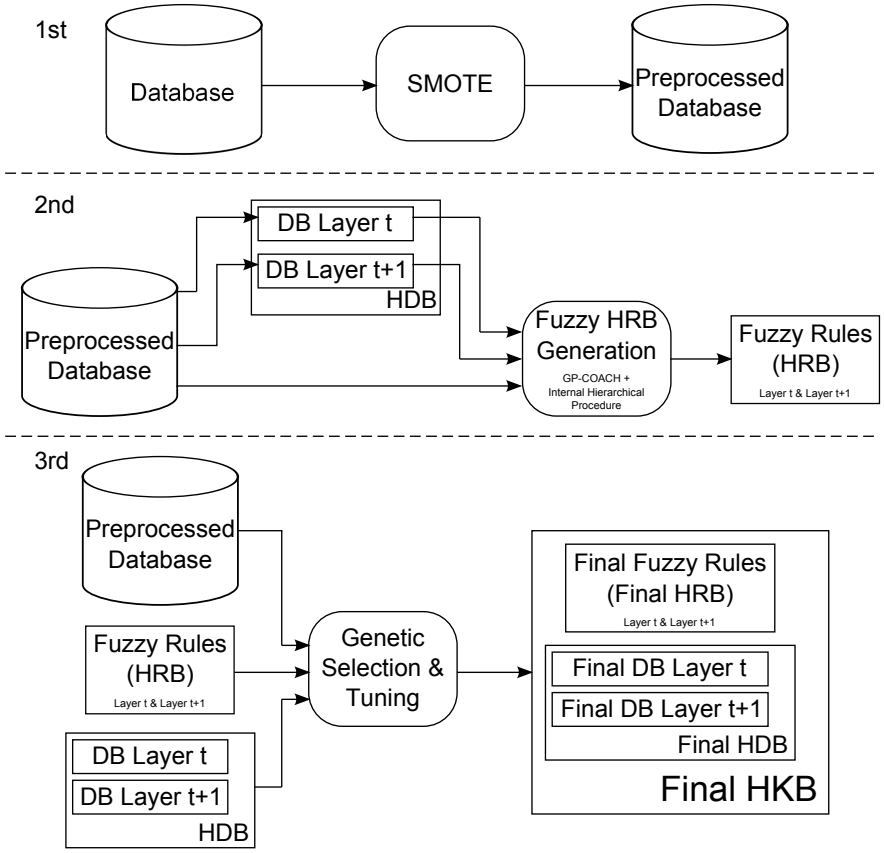


Figure 5: Flowchart of GP-COACH-H

more borderline than others. In this situation, the obtained results are even more definitive as there is a huge gap between the performance of the proposal and the comparison methods. This demonstrates that the proposal is even more effective when confronted with the data intrinsic characteristic themselves.

5.2.2. On the Importance of the Validation Technique for Classification with Imbalanced Datasets: Addressing Covariate Shift when Data is Skewed

The data intrinsic characteristics discernible in the data degrade the performance of classifiers over imbalanced datasets to a further extent than if they were applied to more or less balanced datasets. One of this data intrinsic characteristics is what is known as the dataset shift problem. Dataset shift is defined as the case where training and test data follow different distributions. One of the types of dataset shift is what is called covariate shift, where the input attribute values have different distributions between the training and test sets.

Cross-validation is a technique used for assessing how a classifier will perform when classifying new instances of the task at hand. When a k-fold cross-validation procedure is used, the original sample is randomly partitioned into k subsamples; one of this subsamples is used as test set and the other $k-1$ subsamples will build the training set. However, when a dataset is partitioned in training and test sets, it may induce dataset shift. The DOB-SCV algorithm [MTSH12] is a cross-validation procedure that tries to limit the impact of partition-induced covariate shift and prior-probability

shift.

In this work, we compare the performance of different machine learning methodologies using a standard stratified cross-validation scheme against the cross-validation datasets obtained with the DOB-SCV algorithm. In this manner, we compare how the algorithms behave in a more hostile environment, that is, when more dataset shift is appreciable, and in a more favorable environment when the dataset shift is reduced by a more sensible partitioning method. This methodology enables us to compare the extent of the influence of the dataset shift problem over imbalanced datasets using diverse classification paradigms.

The experimental study developed uses sixty-six imbalanced datasets that range from low imbalanced datasets to highly imbalanced datasets. The methods compared are the C4.5 decision tree [Qui93], the Chi et al's FRBCS [CYP96], the nearest neighbor classifier [CH67], the SMO support vector machine [CV95] and the PDFC classifier [CW03]. These algorithms have been run over the datasets preprocessed with the SMOTE algorithm [CBHK02] so that their results are not biased because of the uneven class distribution.

The results obtained show that there are statistical differences between the usage of the two selected different partitioning methods with only one single run of the partitioning scheme. This indicates the damaging impact that the covariate shift has on imbalanced data, as these differences are not always observed in one run when balanced datasets are compared [MTSH12].

However, these differences are more noticeable in some methods than others. For instance, the C4.5 decision tree is the method that is more affected by the presence of dataset shift which is closely followed by the HFRBCS(Chi) classifier. In the opposite case, we can find the SMO and PDFC methods as the ones that are less affected by the differences in the distribution between the training and test sets.

Furthermore, the experimental study also demonstrates that dataset shift has a damaging effect proportional to the imbalance ratio associated to the corresponding dataset. When the performance of the low imbalanced datasets is contrasted with the performance of the methods for the highly imbalanced datasets, we can observe that the differences detected are greater for this second group of data, and also, that these differences are more stable for the low imbalanced datasets. These results corroborate the initial intuition that dataset shift had a pernicious effect over the skewed class distributions and they encourage the usage of appropriate partitioning methods especially in the imbalanced scenario to avoid undesirable data intrinsic characteristic problems.

The journal articles associated to this part are:

- V. López, A. Fernández, M. J. del Jesus, F. Herrera, A Hierarchical Genetic Fuzzy System Based On Genetic Programming for Addressing Classification with Highly Imbalanced and Borderline Data-sets. Knowledge-Based Systems 38 (2013) 85–104, doi: 10.1016/j.knosys.2012.08.025

- V. López, A. Fernández, F. Herrera, On the Importance of the Validation Technique for Classification with Imbalanced Datasets: Addressing Covariate Shift when Data is Skewed. Information Sciences 257 (2014) 1–13, doi: 10.1016/j.ins.2013.09.038

5.3. A study on the Scalability of FRBCSs for Imbalanced Datasets in the Big Data Scenario

One of the most highlighted trends in the recent years by the information technology industry is what is known as big data. Learning from big data implies the treatment and analysis of datasets with a colossal size. These data collections have some specific features that added up to the enormous amount of information pose a challenge to the performance of standard classifier algorithms.

The data available in big data usually comes from heterogeneous sources which usually introduce some degree of variety to data. Furthermore, this data also has a lot of volatility and variability, is often incomplete and the veracity of the information is questionable. In this situation fuzzy rule-based classification systems are able to provide a model that is able to manage all the uncertainty and ambiguity that is inherent to big data while providing a good trade-off between precision and interpretability.

However, a standard FRBCS that is not adapted to consider the uneven class distribution is not able to provide good classification results for imbalanced datasets. Among the techniques that are able to tackle the imbalanced problem, cost-sensitive learning seems like a sensible choice as it incorporates the misclassification costs into the algorithm design without highly increasing the complexity of the model.

In this work, we propose the usage of a linguistic FRBCS which we have called Chi-FRBCS-BigDataCS. This method is based on the MapReduce framework, one of the most popular approaches towards big data nowadays. The MapReduce model distributes the computation into several independent processing units following two key operations: a Map-function and a Reduce-function.

The Chi-FRBCS-BigDataCS method is based on the original Chi et al's algorithm [CYP96]. The original Chi et al's algorithm is modified to include the misclassification costs of the instances belonging to each class. In order to do so, we modify the computation of the rule weight modifying the original penalized certainty factor so that it consider the misclassification costs.

The classification process for Chi-FRBCS-BigDataCS algorithm is divided in two different MapReduce steps: the building of the model, which describes how the KB is created; and the estimation of the classes for a dataset, which predicts the class for the samples that belong to a big dataset.

The MapReduce procedure associated to the building of the model divides is performed in three different steps: an initial step that computes the DB and the costs associated to each class, and that divides the training set in parts and distributes them to each processing node; a map step that creates a fuzzy rule for each example available in its partition following the Chi et al's method with the new rule weight estimation; and a reduce step that combines the fuzzy rules computed by each map process. The reduce step just adds all the rules to a bigger rule base, however, when equivalent or contradictory rules are encountered, only the rule with the highest rule weight is kept in the final rule base. Figure 6 displays a flowchart describing this building phase.

When the building of the model is finished, the MapReduce method to estimate the class of the examples belonging to a big dataset is initiated. This phase is also divided in several steps: the initial step performs a segmentation of the input dataset into blocks and transfers them to other machines; then, the map step estimates the class for all the examples available in its data partition using the previously built model; finally, the last step aggregates the predictions computed previously as a concatenation of the predictions obtained by each process. This MapReduce procedure is depicted in Figure 7.

The experimental study developed in this work is divided in two parts: a first part that analyzes

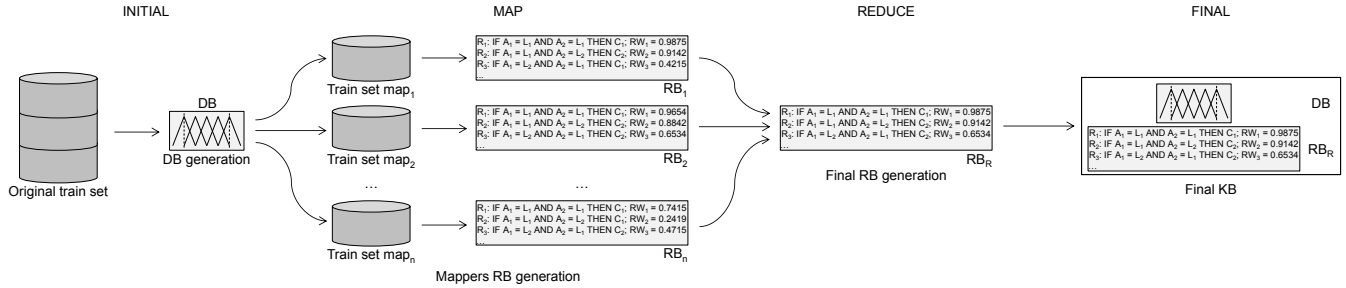


Figure 6: A flowchart of how the building of the KB is organized in Chi-FRBCS-BigData

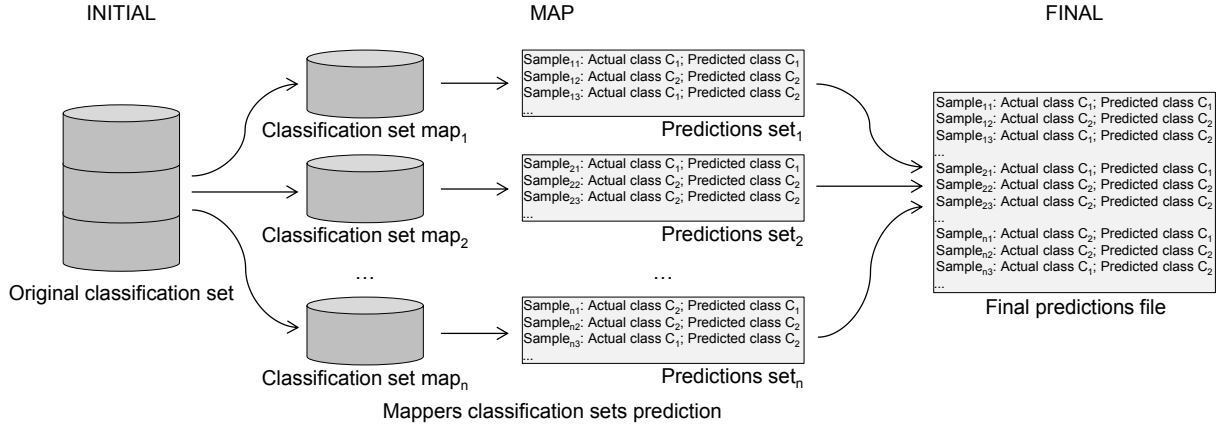


Figure 7: A flowchart of how the classification of a big data classification set is organized in Chi-FRBCS-BigData

the performance of the serial models and a second part that evaluates the performance of the Chi-FRBCS-BigDataCS algorithm over several imbalanced big data cases of study.

To examine the performance of the sequential versions, we select some of the big data cases of study used in this work, and we create reduced versions selecting a percentage of the instances from each class that are later run with the sequential versions of the Chi et al's method (the original one and the cost-sensitive approach). The results obtained show that the sequential versions are only able to provide results for the smaller big data cases of study. For the larger big data cases of study, the sequential versions are only able to provide an answer for some of the reduced versions, up to a 25 % of the samples of each class.

The results obtained for the Chi-FRBCS-BigDataCS approach demonstrate that the MapReduce framework is able of dealing with big data for fuzzy rule-based classification systems. According to the precision of the model (calculated with the AUC measure), the Chi-FRBCS-BigDataCS approach obtains a worse performance than its sequential counterpart. This behavior is clearly noticed when the number of mappers (divisions for parallelization) is increased as the available data to build each rule is smaller and is more difficult to properly describe the original dataset.

When the runtime of the model is examined, a speed gain is observed for the higher values of the number of mappers. Nevertheless, this speed gain is not lineal: the speed gain observed for the smaller values of the number of mappers is higher than the speed gain observed for larger values of the number of mappers. In this manner, it is necessary to further analyze an optimal value for

the number of mappers to find a trade-off between a value small enough to preserve the predictive capacity of the model and a large value that obtains the minimum runtimes.

The journal article associated to this part is:

- V. López, S. del Río, J. M. Benítez, F. Herrera, Cost-Sensitive Linguistic Fuzzy Rule Based Classification Systems under the MapReduce Framework for Imbalanced Big Data. Fuzzy Sets and Systems, doi: 10.1016/j.fss.2014.01.015, ***in press (2014)***

6. Concluding Remarks

The main focus of this PhD dissertation has been to get an insight about classification with imbalanced datasets and its related challenges. Our focus of interest has been the performance of diverse proposals on the topic and the analysis of the data intrinsic characteristics which alter the learning of imbalanced datasets. To analyze these issues we have focused on fuzzy rule-based classification systems as learners because they are effective tools that provide a good trade-off between the precision and interpretability of the models.

In a first step, our aim was to gain a deep understanding about classification with imbalanced datasets and the issues that need to be addressed to improve the performance of methods that are able to address this problem. To complete both objectives, we needed to thoroughly revise the state-of-the-art related to classification with imbalanced data. In doing so, we appreciated that even when numerous proposals had been given to approach the problem, they had not been experimentally compared difficulting the selection of a solution from a practitioner point of view.

In this manner, we decided to perform an extensive analysis of diverse solutions recommended for skewed class distributions. We started performing a comparison between preprocessing techniques and cost-sensitive learning. To do so, we selected several algorithms from diverse classification paradigms, namely, decision trees, support vector machines, fuzzy rule-based classification systems and instance-based learning. The results were not able to find a superiority of one approach over another even when slight differences were found for certain baseline classifiers.

As this first study was not conclusive enough, we decided to extend the previous comparison adding some ensembles of classifiers to the survey, and we extended the number of preprocessing approaches and cost-sensitive learning techniques considered. Furthermore, instead of comparing all the methods alltogether, we decided to compare the diverse methodologies in families comparisons, and only the methods that showed a better performance were selected to evaluate their performance with respect to methods belonging to other families. In general, the proposals showed a more or less similar behavior, where the ensembles of classifiers obtained better results when the base classifier is a weak learner.

The study of the state-of-the-art has not only provided an insight about the approaches that can be used to tackle the problem of imbalanced classification but also it has provided information about what we have called the data intrinsic characteristics. The data intrinsic characteristics are some features that can be appear in the data and that negatively affect the performance of methods in imbalanced datasets. These characteristics can also emerge in balanced datasets, however, their influence in the performance of classifiers in the imbalanced scenario is much more disastrous than in the general case.

This data intrinsic characteristics include the presence of small disjuncts, the lack of density

and information in the training data, the problem of overlapping between the classes, the impact of noisy data in imbalanced domains, the significance of the borderline instances to perform a correct identification of samples that belong to each class and the differences between the training and test data, also known as dataset shift. We have thoroughly discussed how they affect the classification performance in imbalanced data and we have included some experimental results that try to establish a baseline between the impact of each one of this data intrinsic characteristics.

In a second step, we developed some studies that focus on the identified data intrinsic characteristics to improve the performance of classifiers when there is an uneven class distribution.

In the first of this studies, a new hierarchical fuzzy rule-based classification system is proposed to deal with imbalanced problems which we have called GP-COACH-H. This method features two different granularities that are used to better represent each class. In this manner, low granularity rules cover the more general concepts while high granularity rules cover the most specific concepts, traditionally related to the minority class.

These different granularities also allow the model to confront some of the data intrinsic characteristics. The number of generated small disjuncts is lower with this methodology, as it is able to cover small data areas. Furthermore, the overlapping between the classes is better addressed, as the method adapts its behavior in the class frontiers and therefore, it obtains a better class separation. In addition, the borderline samples are better covered because rules with higher granularity are used to properly identify those examples.

The second study devoted to the improvement of methods using the data intrinsic characteristics analyzes the impact of dataset shift over classification with imbalanced datasets. In this case, we observe the dataset shift, and more specifically covariate shift, that is induced by the data partitioning scheme that is traditionally used to validate a new proposal. We compare the performance results obtained using a standard stratified cross-validation procedure with the ones achieved by DOB-SCV, a novel partitioning algorithm which has been proposed precisely to alleviate the addition of covariate shift.

The experimental study associated demonstrated that the partitioning scheme has a strong impact on the performance of classifiers as we included several approaches that represented diverse machine learning paradigms. Furthermore, this influence varies when different learners are used. Dataset shift has also a dissimilar behavior when different degrees of imbalance are considered: for the low imbalanced datasets, the impact of dataset shift is more limited than in the highly imbalanced datasets, where we also observed an elevated variability of results.

In a third step, we decided to explore how skewed class distributions are influenced by one of the latest trends in the information technology industry: Big Data. Big data applications are increasingly becoming the main focus of attention because of the enormous increment of data generation and storage that has taken place in the last years. This situation becomes a challenge when huge amounts of data are processed to extract knowledge because the data mining techniques are not adapted to the new space and time requirements. Furthermore, big data tends to introduce some degree of uncertainty and ambiguity because their data comes from various sources, with different levels of validity and with incomplete information.

To deal with this type of problem, we have proposed the Chi-FRBCS-BigDataCS algorithm, a fuzzy rule-based classification method that is able to deal with imbalanced big data. It is based on the MapReduce framework, one of the most popular approaches nowadays to approach big data problems. As a fuzzy rule-method, it is able to effectively address the vagueness in the data while providing a good performance. Our proposal is based in cost-sensitive learning, which enables it to deal with the uneven class distribution.

The results associated to this study show that it is necessary to specifically address big data problems, as the sequential counterparts are not able to provide results even in some reduced versions of the cases of study considered. However, the developed model performance depends on the number of mappers considered for the experiments. When a high number of mappers is used, the model obtain slow runtimes, however, the performance of the classifier is also affected. If a small number of mappers is considered, then, the classification performance is notably improved, but it comes at the expense of a rise in the runtime spent by the model.

Conclusiones

El principal objetivo de esta Tesis Doctoral ha sido el de profundizar en la clasificación de datos no balanceados y los retos que representa. Nuestro interés se ha centrado en la caracterización del rendimiento de diferentes propuestas acerca del tema y el análisis de las características intrínsecas de los datos que alteran el aprendizaje con datos no balanceados. Para analizar estas cuestiones, nos hemos centrado en la utilización de sistemas de clasificación basados en reglas difusas debido a que son herramientas efectivas que proporcionan un buen equilibrio entre la precisión y la interpetabilidad de los modelos.

En primera aproximación, nuestro objetivo fue conseguir un conocimiento profundo de la clasificación con datos no balanceados y los problemas que deberían resolverse para mejorar el rendimiento de los métodos que consiguen resolver el problema. Para completar ambos objetivos, necesitábamos repasar por completo el estado del arte de la clasificación con datos no balanceados. En el proceso, pudimos apreciar que a pesar de que se habían planteado muchas soluciones para el problema, no se habían comparado experimentalmente, lo que dificultaba la selección de una solución desde un punto de vista práctico.

En este sentido, decidimos realizar un análisis completo de diferentes soluciones recomendadas para distribuciones sesgadas. Comenzamos realizando una comparación entre técnicas de preprocesamiento y aprendizaje sensible al coste. Para ello, seleccionamos varios algoritmos de diferentes paradigmas de clasificación, como son árboles de decisión, máquinas de soporte vectorial, sistemas de clasificación basados en reglas difusas así como aprendizaje basado en instancias. Considerando los resultados obtenidos, no era posible determinar la superioridad de un enfoque frente a otro incluso cuando aparecieron ligeras diferencias para ciertos clasificadores básicos.

Dado que este primer análisis no resultó suficientemente esclarecedor, decidimos extender la comparativa anterior añadiendo algunos grupos de clasificadores al estudio, de forma que aumentamos la cantidad de alternativas para los preprocesadores así como las técnicas de aprendizaje sensible al coste. Además, en lugar de comparar todos los métodos al mismo tiempo, decidimos agruparlos en familias de comparación, de forma que sólo los métodos que demostraron mejor rendimiento se evaluaron con respecto a métodos de otras familias. En general, las propuestas mostraron un comportamiento similar, donde los grupos de clasificadores obtenían mejores resultados al trabajar con clasificadores débiles.

Esta revisión del estado del arte no sólo ha proporcionado una visión más profunda de cómo las propuestas pueden usarse para afrontar el problema de clasificación no balanceada sino que también ha proporcionado información acerca de lo que hemos llamado características intrínsecas de los datos. Las características intrínsecas de los datos son algunas características que pueden estar presentes en los datos y que afectan negativamente el rendimiento de los métodos en datos

no balanceados. Estas características pueden aparecer también en datos balanceados, sin embargo su influencia en el rendimiento de los clasificadores en el caso no balanceado es muchísimo más desastroso que en el caso general.

Las características intrínsecas de los datos incluyen la presencia de pequeños grupos disjuntos, falta de densidad e información en los datos de entrenamiento, el problema del solapamiento entre las clases, el impacto de datos ruidosos en dominios no balanceados, la importancia de las instancias de borde para realizar una correcta identificación de las muestras que pertenecen a cada clase, y las diferencias entre los datos de entrenamiento y de test, también conocido como cambio en conjunto de datos. Hemos analizado en profundidad cómo afectan el rendimiento de la clasificación en datos no balanceados y hemos incluido algunos resultados experimentales que intentan determinar los fundamentos del impacto de cada una de estas características intrínsecas.

En segunda instancia, hemos desarrollado algunos estudios que se centran en las características intrínsecas de los datos para mejorar el rendimiento de clasificadores cuando hay una clase con distribución no uniforme.

Para el primero de estos análisis, se propone un nuevo sistema de clasificación jerárquico basado en reglas difusas para trabajar con problemas no balanceados, que hemos denominado GP-COACH-H. Este método dispone de dos diferentes granularidades que se pueden usar para representar de la manera más fiel posible cada clase. De este modo, las reglas de baja granularidad cubren los conceptos más generales, mientras que las reglas de gran granularidad cubren los conceptos más específicos, relacionados tradicionalmente con las clases minoritarias.

Esta diferenciación de granularidades también permite afrontar algunas de las características intrínsecas de los datos. El número de conjuntos disjuntos generados es menor con esta metodología, ya que es capaz de cubrir pequeñas áreas de datos. Además, el solapamiento entre las clases se maneja mejor, ya que el método adapta su comportamiento en las clases frontera y por lo tanto, consigue una mejor separación de clases. Además, las muestras del borde se interpretan mejor ya que las reglas con granularidad alta se utilizan para identificar adecuadamente esos ejemplos.

El segundo estudio se ha centrado en la mejora de los métodos que utilizan las características intrínsecas de los datos y analiza el impacto del cambio en conjunto de datos para la clasificación con datos no balanceados. En este caso, observamos el cambio, y más específicamente el sesgo de la covarianza, que se induce mediante un esquema de particionado que tradicionalmente se ha usado para validar una nueva propuesta. Comparamos el rendimiento asociado resultante utilizando un procedimiento estratificado estándar de validación cruzada con los resultados alcanzados por DOB-SCV, un nuevo algoritmo de particionamiento que se ha propuesto precisamente para aliviar la presencia del sesgo en la covarianza.

El estudio experimental asociado demostró que el esquema de particionado tiene un importante impacto en el rendimiento de los clasificadores ya que incluimos varios enfoques que representaban diferentes paradigmas de aprendizaje automático. Además, esta influencia varía cuando se utilizan distintos esquemas de aprendizaje. El cambio en conjunto de datos también posee un comportamiento diferenciado cuando se consideran distintos grados de desbalanceo: para los datos con bajo desbalanceo, el impacto del cambio es más limitado que en los datos no balanceados, donde también se ha podido constatar una elevada variabilidad de resultados.

Para el tercer paso, decidimos explorar cuál es la repercusión en las distribuciones de clases con cambio en conjunto de datos de una de las últimas tendencias de la industria de las tecnologías de la información: *Big data*. Las aplicaciones de *big data* se están convirtiendo cada vez más en el foco de atención principal debido el enorme incremento en la generación y almacenamiento de información que ha tenido lugar en los últimos años. Esta situación se convierte en un reto cuando cantidades

ingentes de datos se procesan para la extracción de conocimiento debido a que las técnicas de minería de datos no están adaptadas a los nuevos requerimientos de tiempo y espacio. Además, en *big data*, se tiende a introducir un cierto grado de incertidumbre y ambigüedad ya que los datos proceden de diferentes fuentes, con ciertos niveles de validez y con información incompleta.

Para tratar con este tipo de problemas, hemos propuesto el algoritmo Chi-FRBCS-BigDataCS, un método de clasificación basado en reglas difusas que es capaz de procesar *big data* no balanceado. Se basa en el entorno *MapReduce*, uno de los enfoques más populares de la actualidad para el tratamiento de los problemas de *big data*. Como método basado en reglas difusas, es capaz de resolver de forma efectiva la imprecisión en los datos a la vez que mantiene buen rendimiento. Nuestra propuesta está basada en aprendizaje sensible al coste, que permite manejar las clases con distribuciones no uniformes.

Los resultados asociados a este estudio demuestran que es necesario tratar específicamente los problemas de *big data*, al igual que los correspondientes componentes secuenciales no son capaces de proporcionar los resultados incluso en algunas versiones simplificadas de los casos de estudio considerados. Sin embargo, el rendimiento del modelo desarrollado depende del número de mapeadores considerados para los experimentos. Cuando se utiliza un número elevado de mapeadores, el modelo produce tiempos de ejecución bajos, pero el rendimiento del clasificador se ve afectado. Si se considera un número pequeño de mapeadores, entonces el rendimiento de la clasificación mejora notablemente, pero con un mayor coste en tiempo de ejecución por parte del modelo.

7. Future Work

During the studies developed in this thesis, numerous issues have arisen as interesting paths of research to be further explored.

Extending the modifications based on the data intrinsic characteristics to multi-class imbalanced problems In the literature, there has been little work done in the framework of data sets with multiple imbalanced classes. This opens a wide horizon of possibilities for solving such problems not only with FRBCSs, if not with any type of learning paradigms.

In our case, we are mostly interested in the implementation of various proposals that can help increase the accuracy obtained by the state-of-the-art methods. Introducing operations to deal with the data intrinsic characteristics with problems with more than two classes can end up with models that have better performance values. Moreover, these methods must consider the possibility of building a model that can combine the outputs of small classifiers that are able to better identify minority class instances with respect to larger classes.

Advanced ensembles methods for imbalanced problems In the field of imbalanced datasets, ensembles of classifiers which have been developed in the state-of-the-art have followed the classical ensemble approaches (Boosting and Bagging). These approaches have been combined with preprocessing methods achieving very good results.

However, in the literature the newest methods do not only focus on the traditional ensemble methods but also on some advanced ensemble methods. Therefore, we considered its application to the problem of imbalanced classes. In this manner, we have to find a way suitable for inputting

the pre-processing methods and techniques to address the problem of imbalanced classes in each construction method. To this end, we propose the use of ensembles such as Random Linear Oracle, Rotation Forest or other methods based on projections, among others.

Analyze the interaction of active learning with imbalanced datasets The performance of a predictive model is tightly coupled with the data used during training. In active learning, the model itself plays a hands-on role in the selection of examples for labeling from a large pool of unlabeled examples. It is quite interesting to explore the interaction between active learning and class imbalance, discussing active learning techniques designed specifically for dealing with imbalanced settings, strategies that leverage active learning to overcome the deleterious effects of class imbalance, how extreme class imbalance can prevent active learning systems from selecting useful examples, and alternatives to active learning in these cases.

The design of voting models for ensemble learning algorithms in the context of big data Ensemble learning is one of the most promising areas in machine learning, which is used satisfactorily in many real-world applications. These approaches build a set of classifiers and then classify new data by taking a vote of their predictions. Two of the most representative ensemble learning approaches are bagging and boosting. An important issue in ensemble learning is the technique to combining predictions (or voting scheme) of ensemble classifiers for big data, since it may give different results depending upon different factors. The MapReduce approaches developed in this dissertation have used just a majority voting approach in the Reduce phase to combine the output of the classifiers built in each data partition used by each Map proccess. Therefore, we need to develop the appropriate combination approaches for partial models extracted in a MapReduce framework.

Part II. Publications: Published and Accepted Papers

1. A Study on the Data Intrinsic Characteristics in Classification Problems with Imbalanced Datasets and Analysis of the Behavior of the Techniques from the State-of-the-art

The journal papers associated to this part are:

1.1. Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics

- V. López, A. Fernández, J. G. Moreno-Torres, F. Herrera, Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics. Expert Systems with Applications 39:7 (2012) 6585–6608, doi: 10.1016/j.eswa.2011.12.043

  - Status: **Published**.
  - Impact Factor (JCR 2012): 1.854.
  - Subject Category: Computer Science, Artificial Intelligence. Ranking 31 / 115 (**Q2**).
  - Subject Category: Engineering, Electrical & Electronic. Ranking 56 / 243 (**Q1**).
  - Subject Category: Operations Research & Management Science. Ranking 13 / 79 (**Q1**).

Expert Systems with Applications 39 (2012) 6585–6608

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics

Victoria López [a,*], Alberto Fernández [b], Jose G. Moreno-Torres [a], Francisco Herrera [a]

[a] Dept. of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, 18071 Granada, Spain
[b] Dept. of Computer Science, University of Jaén, 23071 Jaén, Spain

ARTICLE INFO

Keywords:
Classification
Imbalanced datasets
Preprocessing
Cost-sensitive learning
Class overlap
Dataset shift

ABSTRACT

Class imbalance is among the most persistent complications which may confront the traditional supervised learning task in real-world applications. The problem occurs, in the binary case, when the number of instances in one class significantly outnumbers the number of instances in the other class. This situation is a handicap when trying to identify the minority class, as the learning algorithms are not usually adapted to such characteristics.

The approaches to deal with the problem of imbalanced datasets fall into two major categories: data sampling and algorithmic modification. Cost-sensitive learning solutions incorporating both the data and algorithm level approaches assume higher misclassification costs with samples in the minority class and seek to minimize high cost errors. Nevertheless, there is not a full exhaustive comparison between those models which can help us to determine the most appropriate one under different scenarios.

The main objective of this work is to analyze the performance of data level proposals against algorithm level proposals focusing in cost-sensitive models and versus a hybrid procedure that combines those two approaches. We will show, by means of a statistical comparative analysis, that we cannot highlight an unique approach among the rest. This will lead to a discussion about the data intrinsic characteristics of the imbalanced classification problem which will help to follow new paths that can lead to the improvement of current models mainly focusing on class overlap and dataset shift in imbalanced classification.

1. Introduction

One major problem facing data mining is the class imbalance problem (He & Garcia, 2009; Sun, Wong, & Kamel, 2009). It appears in many applications, and is defined as the case where there exists a significant difference between the class prior rates, that is, the probability a particular example belongs to a particular class. The class imbalance is dominant in a high number of real problems including, but not limited to, telecommunications, WWW, finances, ecology, biology, medicine and so on. It must also be stressed that the positive or minority class is usually the one that has the highest interest from the learning point of view and it also implies a great cost when it is not well classified (Elkan, 2001).

A wide number of approaches have been proposed to the imbalanced learning problem that fall largely into two major categories. The first one is data sampling in which the training instances are modified in such a way as to produce a balanced data distribution that allow classifiers to perform in a similar manner to standard classification (Batista, Prati, & Monard, 2004; Chawla, Bowyer, Hall, & Kegelmeyer, 2002). The second one is through algorithmic modification to make base learning methods more attuned to class imbalance issues (Zadrozny & Elkan, 2001). Cost-sensitive learning solutions incorporating both the data and algorithm level approaches assume higher misclassification costs with samples in the rare class and seek to minimize the high cost errors (Ling, Yang, Wang, & Zhang, 2004; Zadrozny, Langford, & Abe, 2003).

Works in imbalanced classification usually focus on the development of new algorithms along one of the categories previously mentioned. However, there is not a study that exhaustively compares solutions from one category to another making difficult the selection of one kind of algorithm when classifying. The aim of this paper is to develop a thorough experimental study to analyze the possible differences between preprocessing techniques and cost-sensitive learning for addressing classification with imbalanced data. In addition, we also present in the comparison a hybrid procedure that combines those two approaches to check whether there is a synergy between them.

In order to analyze the oversampling and undersampling methodologies against cost-sensitive learning approaches, we will use

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: vlopez@decsai.ugr.es (V. López), alberto.fernandez@ujaen.es (A. Fernández), jose.garcia.mt@decsai.ugr.es (J.G. Moreno-Torres), herrera@decsai.ugr.es (F. Herrera).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.12.043

the "Synthetic Minority Over-sampling Technique" (SMOTE) (Chawla et al., 2002) and its variant with the Wilson's Edited Nearest Neighbor (ENN) rule (Wilson, 1972) as they have shown to obtain a very robust behaviour among many different situations (Batista et al., 2004; Fernández, García, del Jesus, & Herrera, 2008). As cost-sensitive methods we study several modifications to well-known classification methods such as C4.5 (Quinlan, 1993), Support Vector Machines (SVMs) (Vapnik, 1998), k-Nearest Neighbor classifier (k-NN) (Cover & Hart, 1967) or Fuzzy Hybrid Genetics-Based Machine Learning (FH-GBML) rule generation algorithm (Ishibuchi & Yamamoto, 2005). The combination of these approaches is carried out through a wrapper classifier (Chawla, Cieslak, Hall, & Joshi, 2008) that uses the aforementioned cost-sensitive techniques with the preprocessing technique obtaining the adequate parameters to perform altogether.

In this work, we focus on imbalanced binary classification problems, having selected a benchmark of 66 problems from KEEL dataset repository[1] (Alcalá-Fdez et al., 2011). We perform our experimental study focusing on the precision of the models using the Area Under the ROC curve (AUC) (Huang & Ling, 2005). This study is carried out using nonparametric tests to check whether there exist significant differences among the obtained results (Demšar, 2006; García & Herrera, 2008).

On the other hand, after comparing these techniques we also want to find what is the source where the difficulties for imbalanced classification emerge. Many other studies on the behavior of several standard classifiers in imbalance domains have shown that significant loss of performance is mainly due to skew of class distributions. However, several investigations also suggest that there are other factors that contribute to such performance degradation, for example, size of the dataset, class imbalance level, small disjuncts, density, and overlap complexity (Japkowicz & Stephen, 2002; Prati & Batista, 2004; Weiss & Provost, 2003). This work focuses on the analysis of two of the most pressing open problems related to data intrinsic characteristics: overlap and dataset shift.

This paper is organized as follows: first, Section 2 presents the problem of imbalanced datasets and the metric we have employed in this context whereas Section 3 describes some ways to tackle the problem: the preprocessing methods used, cost-sensitive classification and a wrapper approach to combine both. Next, Section 4 describes the algorithms we have used in this study, selected benchmark datasets and the configuration of the methods. In Section 5 an analysis of preprocessing techniques versus cost-sensitive learning approaches can be found. Section 6 is devoted to discuss the imbalanced classification problem characteristics that make that problem difficult, analysing the open problems related to data intrinsic characteristics, class overlap and dataset shift. The conclusions of this work can be found in Section 7. Additionally, we include an appendix with the complete tables of results from the experimental study.

2. Imbalanced datasets in classification

In this section, we first introduce the problem of imbalanced datasets and then we present the evaluation metrics for this type of classification problem which differ from usual measures in classification.

2.1. The problem of imbalanced datasets

In some classification problems, the number of instances of every class can be very different. Specifically when facing a dataset with only two classes, the imbalance problem occurs when one class is represented by a large number of examples, while the other is represented by only a few (Chawla, Japkowicz, & Kotcz, 2004).

The problem of imbalanced datasets is extremely significant (Yang & Wu, 2006) because it is implicit in most real world applications, such as very high resolution airborne imagery (Chen, Fang, Huo, & Li, 2011), e-mail foldering (Bermejo, Gámez, & Puerta, 2011) or micro seismic hazards in coal mines (Sikora, 2011), just citing some of them. It is important to point out that the minority class usually represents the concept of interest, for example patients with illnesses in a medical diagnosis problem; whereas the other class represents the counterpart of that concept (healthy patients).

Usually, standard classifier algorithms have a bias towards the majority class, since the rules that predict the higher number of examples are positively weighted during the learning process in favour of the accuracy metric. Consequently, the instances that belong to the minority class are misclassified more often than those belonging to the majority class. Another important issue related to this type of problem is the presence of small disjuncts in the dataset (Weiss & Provost, 2003) and the difficulty most learning algorithms have in detecting those regions. Furthermore, the main handicap in imbalanced datasets is the overlapping between the examples of the positive and the negative class (García, Mollineda, & Sánchez, 2008). These facts are depicted in Fig. 1(a) and (b) respectively.

2.2. Evaluation in imbalanced domains

The measures of the quality of classification are built from a confusion matrix (shown in Table 1) which records correctly and incorrectly recognized examples for each class.

The most commonly used empirical measure, accuracy (1), does not distinguish between the number of correct labels of different classes, which in the framework of imbalanced problems may lead to erroneous conclusions. For example a classifier that obtains an accuracy of 90% in a dataset with a degree of imbalance 9:1, might not be accurate if it does not cover correctly any minority class instance.

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \tag{1}$$

Because of this, instead of using accuracy, more correct metrics are considered. Specifically, from Table 1 it is possible to obtain four metrics of performance that measure the classification quality for the positive and negative classes independently:

- **True positive rate** $TP_{rate} = \frac{TP}{TP+FN}$ is the percentage of positive cases correctly classified as belonging to the positive class.
- **True negative rate** $TN_{rate} = \frac{TN}{FP+TN}$ is the percentage of negative cases correctly classified as belonging to the negative class.
- **False positive rate** $FP_{rate} = \frac{FP}{FP+TN}$ is the percentage of negative cases misclassified as belonging to the positive class.
- **False negative rate** $FN_{rate} = \frac{FN}{TP+FN}$ is the percentage of positive cases misclassified as belonging to the negative class.

One appropriate metric that could be used to measure the performance of classification over imbalanced datasets is the Receiver Operating Characteristic (ROC) curve (Bradley, 1997). In this curve, the tradeoff between the benefits (TP_{rate}) and costs (FP_{rate}) can be visualized, and acknowledges the fact that the capacity of any classifier cannot increase the number of true positives without also increasing the false positives. The Area Under the ROC Curve (AUC) (Huang & Ling, 2005) corresponds to the probability of correctly identifying which of the two stimuli is noise and which is signal plus noise. AUC provides a single-number summary for the performance of learning algorithms.

[1] http://www.keel.es/datasets.php.

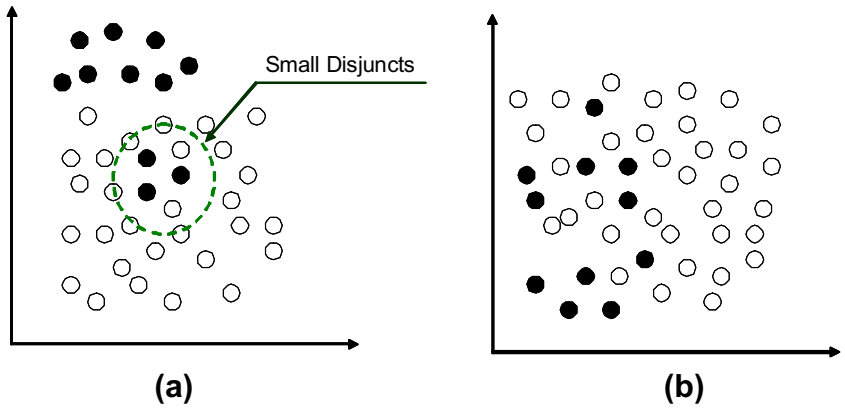


Fig. 1. Example of the imbalance between classes: (a) small disjuncts and (b) overlapping between classes.

Table 1
Confusion matrix for a two-class problem.

	Positive prediction	Negative prediction
Positive class	True positive (TP)	False negative (FN)
Negative class	False positive (FP)	True negative (TN)

The way to build the ROC space is to plot on a two-dimensional chart the true positive rate (Y axis) against the false positive rate (X axis) as shown in Fig. 2. The points (0,0) and (1,1) are trivial classifiers in which the output class is always predicted as negative and positive respectively, while the point (0,1) represents perfect classification. To compute the AUC we just need to obtain the area under the curve as:

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \quad (2)$$

3. Solutions to the class imbalance problem

A large number of approaches have been previously proposed to deal with the class-imbalance problem. These approaches can be categorized in two groups: the internal approaches that create new algorithms or modify existing ones to take the class-imbalance problem into consideration (Barandela, Sánchez, García, & Rangel, 2003; Ducange, Lazzerini, & Marcelloni, 2010; Wu & Chang, 2005; Xu, Chow, & Taylor, 2007) and external approaches that preprocess the data in order to diminish the effect of their class imbalance (Batista et al., 2004; Estabrooks, Jo, & Japkowicz, 2004). Furthermore, cost-sensitive learning solutions incorporating both the data and algorithmic level approaches assume higher misclassification costs with samples in the minority class and seek to minimize the high cost errors (Domingos, 1999; Sun, Kamel, Wong, & Wang, 2007; Zhou & Liu, 2006).

In this section, we first introduce the SMOTE and SMOTE + ENN methods in Section 3.1. Then, cost-sensitive learning is described in Section 3.2. Finally, Section 3.3 presents a framework to automatically detect a threshold for preprocessing using an underlying algorithm, in this case, a cost-sensitive approach.

3.1. Preprocessing imbalanced datasets. The SMOTE and SMOTE + ENN algorithms

As mentioned before, applying a preprocessing step in order to balance the class distribution is an effective solution to the imbalanced dataset problem (Batista et al., 2004). Specifically, in this work we have chosen an oversampling method which is a well-known reference in the area: the SMOTE algorithm (Chawla et al., 2002) and a variant called SMOTE + ENN (Batista et al., 2004) as they have been shown to present a very robust behavior among many different situations (Batista et al., 2004; Fernández et al., 2008).

In this approach, the positive class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen. This process is illustrated in Fig. 3, where x_i is the selected point, x_{i1} to x_{i4} are some selected nearest neighbors and r_1 to r_4 the synthetic data points created by the randomized interpolation.

Synthetic samples are generated in the following way: take the difference between the feature vector (sample) under consideration and its nearest neighbor. Multiply this difference by a random number between 0 and 1, and add it to the feature vector under consideration. This causes the selection of a random point

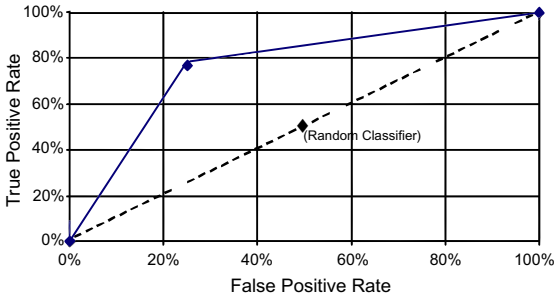


Fig. 2. Example of an ROC plot. Two classifiers are represented: the solid line is a good performing classifier whereas the dashed line represents a random classifier.

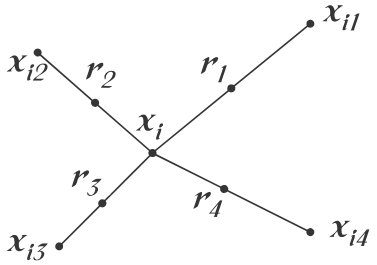


Fig. 3. An illustration of how to create the synthetic data points in the SMOTE algorithm.

along the line segment between two specific features. This approach effectively forces the decision region of the minority class to become more general. An example is detailed in Fig. 4.

In short, its main feature is to form new minority class examples by interpolating between several minority class examples that lie together. Thus, the overfitting problem is avoided and causes the decision boundaries for the minority class to spread further into the majority class space.

Nevertheless, class clusters may be not well defined in cases where some majority class examples invade the minority class space. The opposite can also be true, since interpolating minority class examples can expand the minority class clusters, introducing artificial minority class examples too deeply into the majority class space. Inducing a classifier in such a situation can lead to overfitting. For this reason we also consider a hybrid approach in this work, "SMOTE + ENN", where the Wilson's ENN rule (Wilson, 1972) is used after the SMOTE application to remove from the training set any example misclassified by its three nearest neighbors.

3.2. Cost-sensitive learning

Cost-sensitive learning takes into account the variable cost of a misclassification of the different classes (Domingos, 1999; Zadrozny et al., 2003). A cost matrix codifies the penalties of classifying examples of one class as a different one. Let $C(i,j)$ be the cost of predicting an instance of class i as class j; with this notation $C(+,-)$ is the cost of misclassifying an instance of the positive class as if it was negative and $C(-,+)$ is the cost of the opposite case.

When dealing with imbalanced problems it is usually more important to recognize the positive instances rather than the negative ones. Therefore, the cost of misclassifying a positive instance is higher than the cost of misclassifying a negative one ($C(+,-) > C(-,+)$). As a classical example, the reader may refer to a diagnosis problem in which it is often less dangerous to obtain a false positive than a false negative.

Three main general approaches have been proposed to deal with cost-sensitive problems:

1. Methods based on modifying the training data. The most popular technique lies in resampling the original class distribution of the training dataset according to the cost decision matrix by means of undersampling/oversampling, modifying decision thresholds or assigning instance weights. These modifications have shown to be effective and can also be applied to any cost insensitive learning algorithm (Zadrozny et al., 2003; Zhou & Liu, 2006).
2. Methods that change the learning process in order to build a cost-sensitive classifier, for example, in the context of decision tree induction, the tree-building strategies are adapted to minimize the misclassification costs. The cost information is used to: (1) choose the best attribute to split the data Ling et al. (2004) and Riddle, Segal, and Etzioni (1994); and (2) determine whether a subtree should be pruned Bradford, Kunz, Kohavi, Brunk, and Brodley (1998).
3. Methods based on the Bayes decision theory that assign instances to the class with minimum expected cost. For example, a typical decision tree for a binary classification problem assigns the class label of a leaf node depending on the majority class of the training samples that reach the node. A cost-sensitive algorithm assigns the class label to the node that minimizes the classification cost Domingos (1999) and Zadrozny and Elkan (2001).

```
Consider a sample (6,4) and let (4,3) be its nearest neighbor.
(6,4) is the sample for which k-nearest neighbors
are being identified and (4,3) is one of its k-nearest neighbors.
Let: f1_1 = 6 f2_1 = 4, f2_1 - f1_1 = -2
f1_2 = 4 f2_2 = 3, f2_2 - f1_2 = -1
The new samples will be generated as
(f1',f2') = (6,4) + rand(0-1) * (-2,-1)
rand(0-1) generates a random number between 0 and 1.
```

Fig. 4. Example of the SMOTE application.

Cost-sensitive learning supposes that there is a cost matrix available for the different type of errors. However, given a dataset, this matrix is not usually given Sun et al. (2007, 2009).

3.3. Hybridization. Automatically countering imbalance

The different solutions used to deal with the imbalanced problem have been presented in the previous subsections. So the question now is "Can we use both techniques together and achieve better results?"

Cost-sensitive learning algorithms associate high misclassification costs to positive instances which bias the search towards the positive class. If the cost associated to positive instances is too high or if the specific cost-sensitive algorithm is easily biased towards the positive class, we can observe that the decision region generated by the algorithm is far away from those instances. Therefore, we need to bias those algorithms in a way that pushes the boundary towards the positive instances, but still classifies correctly both classes. If the positive instances are sparse, a case that is likely to occur in imbalanced datasets, then the boundary may not have the proper shape.

On the other hand, preprocessing methods try to balance class distributions to let the standard classifier algorithms accomplish similar results to their performance in a balanced data scenario. There is a diversity of preprocessing methods with a behavior focused on generating new samples, removing some of the samples or carrying out both operations jointly. Nevertheless, these methods can fail due to the loss of information produced when we delete samples that define our decision boundaries or when we create examples that introduce noise to the classifier.

Regarding cost-sensitive learning classifiers, a way to avoid biasing towards positive instances without modifying the algorithm itself lies in the creation of a few positive instances or the deletion of some negative examples. This causes a more balanced data distribution which means that the misclassification costs associated to positive instances will also be lower thus making the search process less biased. In addition, since we are using a cost-sensitive classifier we do not need to apply a preprocessing procedure to balance the distribution because cost-sensitive learners are able to learn in imbalanced conditions, therefore, the resampling stage is quicker than using only a preprocessing approach and the whole learning process is sped up, especially when the base classifier efficiency deeply depends on the number of instances.

We can find some works related to this idea. For example, Akbani, Kwek, and Japkowicz (2004) propose a solution with support vector machines where they integrate a cost-sensitive support vector machine (Veropoulos, Campbell, & Cristianini, 1999) with the SMOTE technique of oversampling the minority instances (Chawla et al., 2002). With this behavior they manage to push the boundary away from the positive instances (cost-sensitive learning) and make the boundary better defined (because of the denser positive instance distribution).

Due to the previous facts we aim to develop a procedure to integrate the cost-sensitive learning and preprocessing approaches into one. Chawla et al. (2008) propose a wrapper paradigm that discovers the amount of resampling needed for a dataset based

on optimizing evaluation functions which can include the cost associated to the classification. This wrapper infrastructure applies cross-validation to first discover the best amounts of undersampling and oversampling, applies the preprocessing algorithms with the amounts estimated and finally runs the algorithm used over the preprocessed dataset.

Obviously, searching the entire space of undersampling and SMOTE combinations can quickly become intractable, so the search procedure must be fine-tuned. This strategy removes the "excess" examples of the majority classes, which reduces the size of the training dataset. This also makes learning time more manageable. SMOTE is used to add synthetic examples of the minority classes and increase the generalization performance of the classifier over the minority classes. Fig. 5 shows the algorithm procedure.

The estimation is done over a training and a test set. The training data is split into five partitions for an internal five-fold cross-validation. The wrapper applies this independent validation stage to each fold to discover the appropriate percentages of sampling for a given method and classifier combination. Once these percentages are discovered, the classifier is re-learned on the original training fold using the discovered percentages and tested on the corresponding testing fold.

The undersampling estimation starts with no undersampling for all majority classes and obtains baseline results on the training data. Then it traverses through the search space of undersampling percentages in decrements of *Sample Decrement*, in a greedy iterative fashion, to increase performance over the minority classes without sacrificing performance on the majority class.

The oversampling algorithm evaluates different amounts of SMOTE at steps of *Sample Increment* (percentage of the number of examples from the minority class that will be generated in each step). This is a greedy search, and at each step the new performance estimates become the new baseline. That is, the initial baseline is the performance obtained via the Wrapper Undersample. If SMOTE = *Sample Increment* improves the performance over that baseline by some margin *Increment Min*, then the performance achieved at SMOTE = *Sample Increment* becomes the new baseline. The amount of SMOTE is then incremented by *Sample Increment*, and another evaluation is performed to check if the performance increase at new SMOTE amount is at least greater than *Increment Min*. This process repeats, greedily, until no performance gains are observed.

However, there is an important caveat to the search to avoid being trapped in a local maximum. If the average does not improve by *Increment Min* we have to verify that we have not settled on a local maximum. In order to do so, we look ahead some more steps at increasing amounts of SMOTE. If the look-ahead does not result in an improvement in performance, then the amount of SMOTE is reset to the value discovered prior to the look-ahead. This is done to allow SMOTE to introduce additional examples with the aim of improving performance. However, if the addition of examples does not help, then we go back to using the lesser amount of SMOTE discovered prior to the look-ahead.

We can use different measures to evaluate the performance of the classifier to estimate the sampling parameters. Since we are using cost-sensitive learning algorithms as base classifiers a logical evaluation criteria is the cost itself. Cost is calculated as shown in Eq. (3) when we assume $C(+|+) = C(-|-) = 0$ (as it is usual in imbalanced classification).

$$cost = FNrate \cdot C(-|+) + FPrate \cdot C(+|-) \tag{3}$$

4. Experimental framework

In this section, we first introduce the algorithms which are included in the study (Section 4.1). Next, we provide details of the imbalanced problems chosen for the experimentation and the configuration parameters of the methods (Sections 4.2 and 4.3). Finally, we present the statistical tests applied to compare the results obtained with the different classifiers (Section 4.4).

4.1. Algorithms selected for the study

This section presents the description of the state of the art algorithms of four different classification paradigms selected for our study. For each paradigm we outline the base classifier commonly used in general classification problems and the cost-sensitive learning version associated to that classifier.

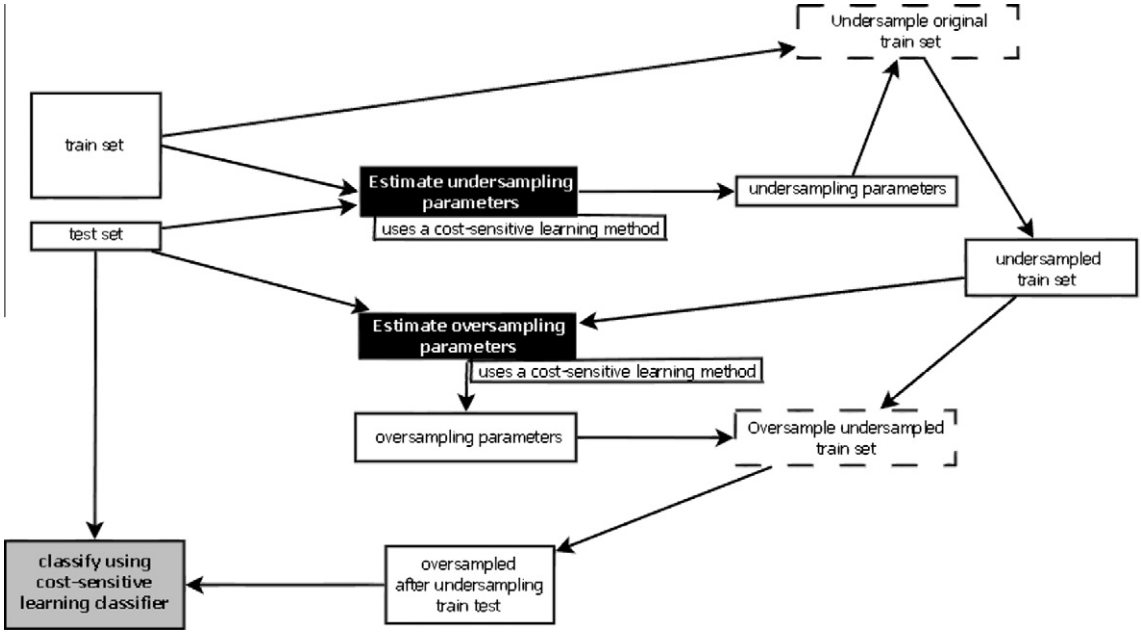


Fig. 5. Illustration on the wrapper undersample SMOTE algorithm. Dashed lines means resampling actions, black boxes represent the parameters estimation and the final result is in grey.

4.1.1. Decision trees

Decision trees use simple knowledge representation to classify examples into a finite number of classes. In a typical setting, the tree nodes represent the attributes, the edges represent the possible values for a particular attribute, and the leaves are assigned with class labels. Classifying a test sample is straightforward once a decision tree has been constructed. An object is classified by following paths from the root node through the tree to a leaf, taking the edges corresponding to the values of attributes.

C4.5 decision tree. C4.5 (Quinlan, 1993) is a decision tree generating algorithm. It induces classification rules in the form of decision trees from a set of given examples. The decision tree is constructed top-down using the normalized information gain (difference in entropy) that results from choosing an attribute for splitting the data. The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurs on the smaller sublists.

Cost-sensitive C4.5 decision tree. The cost-sensitive C4.5 decision tree (C4.5CS) (Ting, 2002) is a method to induce cost-sensitive trees that seeks to minimize the number of high cost errors and, as a consequence of that, leads to minimization of the total misclassification costs in most cases.

The method changes the class distribution such that the tree induced is in favor of the class with high weight/cost and is less likely to commit errors with high cost. Specifically, the computation of the split criteria for C4.5 (normalized information gain) is modified to take into account the *a priori* probabilities according to the number of samples for each class.

C4.5CS modifies the weight of an instance proportional to the cost of misclassifying the class to which the instance belonged, leaving the sum of all training instance weights still equal to N. Let $C(j)$ be the cost of misclassifying a class j instance; the weight of a class j instance can be computed as

$$w(j) = C(j)\frac{N}{\sum_i C(i)N_i} \tag{4}$$

such that the sum of all instance weights is $\sum_j w(j)N_j = N$.

The standard greedy divide-and-conquer procedure for inducing minimum error trees can then be used without modification, except that $W_j(t)$ is used instead of $N_j(t)$ in the computation of the test selection criterion in the tree growing process and the error estimation in the pruning process. That $W_j(t)$ is the result of weighting the initial number of instances from a class with the weight computed in Eq. (4): $W_j(t) = w(j) \cdot N_j(t)$ Thus, both processes are affected due to this change.

This modification effectively converts the standard tree induction procedure that seeks to minimize *the number of errors*, regardless of cost, to a procedure that seeks to minimize *the number of errors with high weight or cost*. To classify a new instance, C4.5CS predicts the class which has the maximum weight at a leaf, as in C4.5.

C4.5CS also introduces another optional modification that alters the usual classification process after creating the decision tree. Instead of classifying using the minimum error criteria, it is advisable to classify using the expected misclassification cost in the last part of the classification procedure. The expected misclassification cost for predicting class i with respect to the instance x is given by

$$EC_i(x) \propto \sum_j W_j(t(x))cost(i,j) \tag{5}$$

where $t(x)$ is the leaf of the tree that instance x falls into and $W_j(t)$ is the total weight of class j training instances in node t.

To classify a new instance x using a minimum error tree with the minimum expected cost criterion, $EC_i(x)$ is computed for every class. The instance x is assigned to class i with the smallest value for $EC_i(x)$; that is, $EC_i(x) < EC_{i'}(x)$ for all $i' \neq i$.

4.1.2. Support vector machines

SVMs are one of the binary classifiers based on maximum margin strategy introduced by Vapnik and Lerner (1963). Originally, SVMs were designed for linear two-class classification with margin, where margin means the minimal distance from the separating hyperplane to the closest data points. SVMs seek an optimal separating hyperplane, where the margin is maximal. The solution is based only on those data points at the margin. These points are called as support vectors. The linear SVMs have been extended to nonlinear examples when the nonlinear separated problem is transformed into a high dimensional feature space using a set of nonlinear basis functions. However, the SVMs are not necessary to implement this transformation to determine the separating hyperplane in the possibly high dimensional feature space. Instead, a kernel representation can be used, where the solution is written as a weighted sum of the values of a certain kernel function evaluated at the support vectors.

Soft margin SVM. In 1995, Cortes and Vapnik suggested a modified maximum margin idea that allows for mislabeled examples (Cortes & Vapnik, 1995; Vapnik, 1998). If there exists no hyperplane that can split the "yes" and "no" examples, the Soft Margin method will choose a hyperplane that splits the examples as cleanly as possible, while still maximizing the distance to the nearest cleanly split examples. The method introduces slack variables, ξ_i, which measure the degree of misclassification of the datum x_i.

Cost-sensitive SVM. The cost-sensitive SVM (SVMCS) (Veropoulos et al., 1999) is a modification of the soft-margin support vector machine. We need to bias SVM in a way that will push the boundary away from the positive instances using different error costs for the positive (C^+) and negative (C^-) classes. Specifically, the change implies a new optimization function

$$\min_{\mathbf{w},\xi,b}\max_{\boldsymbol{\alpha},\boldsymbol{\beta}}\left\{\frac{1}{2}\|\mathbf{w}\|^2 + C^+\sum_{\{i|y_i=+1\}}^{n_+}\xi_i + C^-\sum_{\{j|y_j=-1\}}^{n_-}\xi_j - \sum_{i=1}^{n}\alpha_i[y_i(\mathbf{w}\cdot\mathbf{x_i}-b) - 1+\xi_i] - \sum_{i=1}^{n}\beta_i\xi_i\right\} \tag{6}$$

The constraints on α_i then become:

$$0 \leqslant \alpha_i \leqslant C^+ \quad \text{if } y_i = +1 \tag{7}$$

and

$$0 \leqslant \alpha_i \leqslant C^- \quad \text{if } y_i = -1 \tag{8}$$

Furthermore, $\xi_i > 0$ only when $\alpha_i = C$. Therefore non-zero errors on positive support vectors will have larger α_i while non-zero errors on negative support vectors will have smaller α_i. The net effect is that the boundary is pushed more towards the negative instances.

4.1.3. Fuzzy rule based classification system learning methods

A fuzzy rule based classification system (FRBCS) has two main components: the inference system and the knowledge base. In a linguistic FRBCS, the knowledge base is composed of a rule base, constituted by a set of fuzzy rules, and the data base that stores the membership functions of the fuzzy partitions associated to the input variables.

In this work we use fuzzy rules of the following form for our FRBCSs:

$$\begin{aligned}\text{Rule } R_j:\ &\text{If } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{jn}\\ &\text{then Class} = C_j \text{ with } RW_j\end{aligned} \tag{9}$$

where R_j is the label of the jth rule, $x = (x_1,\ldots,x_n)$ is an n-dimensional pattern vector, A_{ji} is an antecedent fuzzy set, C_j is a class label, and RW_j is the rule weight (Ishibuchi & Nakashima, 2001). We use triangular membership functions as fuzzy partitions associated to the input variables. To compute the rule weight, many alternatives

have been proposed, although we have considered as a good choice the use of the heuristic method known as the Penalized Certainty Factor (PCF) Ishibuchi and Yamamoto (2005):

$$\mathrm{PCF}_j = \frac{\sum_{x_p \in C_j} \mu_{A_j}(x_p) - \sum_{x_p \notin C_j} \mu_{A_j}(x_p)}{\sum_{p=1}^{m} \mu_{A_j}(x_p)} \tag{10}$$

where x_p is the pth example of the training set, C_j is the consequent class of rule j and $\mu_{A_j}(x_p)$ is the membership degree of the example with the antecedents of the rule.

Fuzzy hybrid genetic based machine learning rule generation algorithm. In order to build the rule base, we have chosen the FH-GBML algorithm (Ishibuchi, Yamamoto, & Nakashima, 2005), a proposal that presents a good behaviour in both standard and imbalanced classification (Fernández, del Jesús, & Herrera, 2010; Luengo, Fernández, García, & Herrera, 2011).

The FH-GBML method consists of a Pittsburgh approach where each rule set is handled as an individual. It also contains a Genetic Cooperative-Competitive Learning (GCCL) approach (an individual represents a unique rule), which is used as a kind of heuristic mutation for partially modifying each rule set. This method uses standard fuzzy rules with rule weights (Ishibuchi & Yamamoto, 2005) where each input variable x_i is represented by a linguistic term or label. The system defines 14 possible linguistic terms for each attribute as well as a special "do not care" as an additional linguistic term.

In the learning process, N_{pop} rule sets are created by randomly selecting N_{rule} training patterns. Then, a fuzzy rule from each of the selected training patterns is generated by probabilistically choosing an antecedent fuzzy set from the 14 candidates $\left(P_{donot\ care}(B_k) = \frac{\mu_{B_k}(x_{pi})}{\sum_{j=1}^{14} \mu_{B_j}(x_{pi})}\right)$ and each antecedent fuzzy set of the generated fuzzy rule is replaced with don't care using a pre-specified probability $P_{do\ not\ care}$.

N_{pop} −1 rule sets are generated by selection, crossover and mutation in the same manner as the Pittsburgh-style algorithm. Next, with a pre-specified probability, a single iteration of the Genetic Cooperative–Competitive-style algorithm is applied to each of the generated rule sets.

Finally, the best rule set is added to the current population in the newly generated ($N_{pop} - 1$) rule sets to form the next population and, if the stopping condition is not satisfied, the genetic process is repeated again. Classification is performed following the fuzzy reasoning method of the winning rule.

Cost-sensitive fuzzy hybrid genetic based machine learning rule generation algorithm. The FH-GBML-CS (Fuzzy Hybrid Genetics-Based Machine Learning Cost-Sensitive) algorithm (López, Fernández, & Herrera, 2010) is a modification of the FH-GBML original algorithm. The main goal of FH-GBML-CS is to obtain a FRBCS that is able to consider the different costs associated to misclassification of some of its samples during the building process of the RB. To achieve that purpose an algorithmic level solution is used, modifying the original behaviour of the FH-GBML algorithm in some of its steps:

- *Adaptation of the fitness function of the Pittsburgh approach*. Instead of using the number of correctly classified training examples FH-GBML-CS tries to minimize the misclassification cost: $FN_{rate} \cdot C(-,+) + FP_{rate} \cdot C(+,-)$.
- *Modifications in the computation of the rule weight*. The PCF heuristic has been adapted to cost-sensitive learning building the Cost-Sensitive Penalized Certainty Factor (CS-PCF) which is used in FH-GBML-CS to compute the rule weight:

$$\mathrm{CS-PCF}_j = \frac{\sum_{x_p \in C_j} \mu_{A_j}(x_p) \times \mathrm{Cs}_p}{\sum_{p=1}^{m} \mu_{A_j}(x_p) \times \mathrm{Cs}_p} - \frac{\sum_{x_p \notin C_j} \mu_{A_j}(x_p) \times \mathrm{Cs}_p}{\sum_{p=1}^{m} \mu_{A_j}(x_p) \times \mathrm{Cs}_p} \tag{11}$$

where Cs_p is the misclassification cost of an example from class p.

- *Different class label choice for the rule*. Instead of selecting the class considering only the highest compatibility the class with the highest *compatibility* · *cost* is chosen.

4.1.4. Lazy learning

Lazy learning is a set of methods in which generalization beyond the training data is delayed until a query is made to the system, as opposed to in eager learning, where the system tries to generalize the training data before receiving queries.

The main advantage gained in employing a lazy learning method is that the target function will be approximated locally, such as in the k-NN algorithm. Because the target function is approximated locally for each query to the system, lazy learning systems can simultaneously solve multiple problems and deal successfully with changes in the problem domain.

K-nearest neighbor algorithm. k-NN (Cover & Hart, 1967) finds a group of k instances in the training set that are closest to the test pattern. Given a test sample, the algorithm computes the distance (or similarity) between the test sample and all of the training samples to determine its k-nearest neighbors. The class of the test sample is decided by the most abundant class within the k-nearest samples.

Cost-sensitive k-nearest neighbor algorithm. Cost-sensitive k-NN algorithm (Hand & Vinciotti, 2003) is a cost-sensitive learning version of k-NN based on Bayes risk theory to assign each sample to its lowest risk class.

Let the cost of misclassifying a class i case be c_i. Now, if points at x are assigned to class 1, the loss at x is $c_0 p(0|x)$. Similarly, if points at x are assigned to class 0, the loss at x is $c_1 p(1|x)$. The minimum loss at x is thus achieved by assigning points at x to class 1 if $c_0 p(0|x) < c_1 p(1|x)$ and to class 0 otherwise. This is equivalent to the condition

$$p(1|x) > c_0/(c_0 + c_1) \tag{12}$$

Without loss of generality we will rescale the costs so that $(c_0 + c_1) = 1$, so that the classification rule becomes "Assign points at x to class 1 when $p(1|x) > c_0$ and to class 0 otherwise".

Nearest neighbor methods estimate the $p(i|x)$ by the proportion of class i points amongst the k nearest neighbors to the point x to be classified. This requires a choice of a distance metric and a choice of the parameter k.

To sum up, the cost-sensitive k-NN classification rule assigns a point with measurement vector x to class 1 if $k_1/k > c_0$, and otherwise to class 0, where k_1 is the number of class 1 points amongst the k design set points closest to x.

4.1.5. Summary of the different schemes selected for the experimental study

In this work, we test several combinations of preprocessing and cost-sensitive learning with the classification algorithms from each paradigm described in this section. Specifically, the schemes used can be arranged into three categories:

1. Oversampling approaches to balance the data distribution before applying the algorithm which were described in Section 3.1.
2. Cost-sensitive learning methods which take into consideration costs. The methods used are specific versions that come from the original non-balanced algorithms. These algorithm versions have been described in this section.
3. Application of the hybrid methodology that combines cost-sensitive learning and preprocessing: a methodology to automatically countering imbalance using cost was described in Section 3.3. We use different combinations of algorithms to evaluate the performance of the methodology.

Table 2
Acronyms used to designate the different algorithm variations used in the experimental study.

Acronym	Version description
None	The original classifier that names the algorithm family
SMOTE	The original classifier that names the algorithm family applied to a dataset preprocessed with the SMOTE algorithm
SENN	The original classifier that names the algorithm family applied to a dataset preprocessed with the SMOTE + ENN algorithm
CS	The cost-sensitive version of the original classifier from the corresponding algorithm family which was explained in the previous section
Wr_SMOTE	Version of the Wrapper routine described in the previous section that uses as main algorithm the cost-sensitive version of the algorithm family and only performs the oversampling step with the SMOTE algorithm
Wr_US	Version of the Wrapper routine described in the previous section that uses as main algorithm the cost-sensitive version of the algorithm family, performs the undersampling step with a random undersampling algorithm and the oversampling step with the SMOTE algorithm
Wr_SENN	Version of the Wrapper routine described in the previous section that uses as main algorithm the cost-sensitive version of the algorithm family and only performs the oversampling step with the SMOTE + ENN algorithm

For the sake of clarity, Table 2 indicates a list of acronyms used to identify the different algorithm versions for each paradigm and a brief description of each one of them.

4.2. Datasets and data partitions

In order to analyze the preprocessing approach against the cost-sensitive learning strategy, we have selected 66 datasets from the KEEL dataset repository[2] (Alcalá-Fdez et al., 2011).

In the specialized literature, researchers usually manage all imbalanced datasets as a whole (Barandela et al., 2003; Batista et al., 2004; Chen, Chen, Hsu, & Zeng, 2008). In this work we sort the different datasets according to their degree of imbalance using the imbalance ratio (IR) (Orriols-Puig & Bernadó-Mansilla, 2009), which is defined as the ratio of the number of instances of the majority class and the minority class.

The datasets are summarized in Table 3, where we denote the number of examples (#Ex.), number of attributes (#Atts.), class name of each class (positive and negative), class distribution and IR.

To develop the different experiments we consider a *5-folder cross-validation model*, i.e., five random partitions of data with a 20% and the combination of 4 of them (80%) as training and the remaining one as test. For each dataset we consider the average results of the five partitions. The datasets used in this study use the partitions provided by the repository in the imbalanced classification dataset section.[3]

4.3. Parameters

The configuration parameters for the base classifiers are shown in Table 4. The selected values are common for all the versions of the algorithm in the same family to maintain an experimental scenario on equal terms. On the other hand, the parameters for the preprocessing methods used in this study are presented in Table 5. Finally, Table 6 points out the parameters for the implementation of the wrapper routine. All these values were selected according to the recommendation of the corresponding authors of each algorithm, which is the default parameters' setting included in the KEEL software (Alcalá-Fdez et al., 2008).

The only ad-hoc parameter value is the k parameter of nearest neighbors. We have set that value to $k = 3$ instead of $k = 1$ which is the usual approach because the cost-sensitive k-NN used in this study achieves an identical performance for 1-NN and 1-NNCS.

Furthermore, we have to identify the misclassification costs associated to the positive and negative class for the cost-sensitive learning versions. If we misclassify a positive sample as a negative one the associated misclassification cost is the IR of the dataset ($C(+,-) = IR$) whereas if we misclassify a negative sample as a positive one the associated cost is 1 ($C(-,+) = 1$). The cost of classifying correctly is 0 ($C(+,+) = C(-,-) = 0$) because guessing the correct class should not penalize the built model.

Although we acknowledge that the tuning of the parameters for each method on each particular problem could lead to better results, we chose to maintain a baseline performance of each method as the basis for comparison. Since the experimental study is focused in the performance of methods from the same family, our hypothesis is that methods that win on average on all problems would also win if a better setting was used. Furthermore, in a framework where no method is tuned, winner methods tend to correspond to the most robust learners, which is also a desirable characteristic.

4.4. Statistical tests for performance comparison

Statistical analysis needs to be carried out in order to find significant differences among the results obtained by the studied methods (García, Fernández, Luengo, & Herrera, 2009). We consider the use of non-parametric tests, according to the recommendations made in Demšar (2006), García and Herrera (2008), García et al. (2009), García, Fernández, Luengo, and Herrera (2010) where a set of simple, safe and robust non-parametric tests for statistical comparisons of classifiers is presented. These tests are used due to the fact that the initial conditions that guarantee the reliability of the parametric tests may not be satisfied, causing the statistical analysis to lose credibility (Demšar, 2006).

Since the study is split in parts comparing a group of algorithms, we use statistical tests for multiple comparisons. Specifically, we use the Iman–Davenport test (Sheskin, 2006) to detect statistical differences among a group of results and the Shaffer post-hoc test (Shaffer, 1986) in order to find out which algorithms are distinctive among an $n \times n$ comparison.

The post-hoc procedure allows us to know whether a hypothesis of comparison of means could be rejected at a specified level of significance α, which we set to 95% in our study. However, it is very interesting to compute the p-value associated with each comparison, which represents the lowest level of significance of a hypothesis that results in a rejection. In this manner, we can know whether two algorithms perform significantly differently and to what degree.

Furthermore, we consider the average ranking of the algorithms in order to show graphically how good a method is with respect to its partners. This ranking is obtained by assigning a position to each algorithm depending on its performance for each dataset. The algorithm which achieves the best accuracy in a specific dataset will have the first ranking (value 1); then, the algorithm with the second best accuracy is assigned rank 2, and so forth. This task is carried out for all datasets and finally an average ranking is computed as the mean value of all rankings.

These tests are suggested in the studies presented by Demšar (2006), García and Herrera (2008), and García et al. (2009), where their use in the field of machine learning is strongly recommended.

[2] http://www.keel.es/datasets.php.
[3] http://www.keel.es/imbalanced.php.

Table 3
Summary of imbalanced datasets.

Datasets	# Ex.	# Atts.	Class (−,+)	%Class (−;+)	IR
Glass1	214	9	(build-win-non_float-proc; remainder)	(35.51,64.49)	1.82
Ecoli0vs1	220	7	(im; cp)	(35.00,65.00)	1.86
Wisconsin	683	9	(malignant; benign)	(35.00,65.00)	1.86
Pima	768	8	(tested-positive; tested-negative)	(34.84,66.16)	1.90
Iris0	150	4	(Iris-Setosa; remainder)	(33.33,66.67)	2.00
Glass0	214	9	(build-win-float-proc; remainder)	(32.71,67.29)	2.06
Yeast1	1484	8	(nuc; remainder)	(28.91,71.09)	2.46
Vehicle1	846	18	(Saab; remainder)	(28.37,71.63)	2.52
Vehicle2	846	18	(Bus; remainder)	(28.37,71.63)	2.52
Vehicle3	846	18	(Opel; remainder)	(28.37,71.63)	2.52
Haberman	306	3	(Die; Survive)	(27.42,73.58)	2.68
Glass0123vs456	214	9	(non-window glass; remainder)	(23.83,76.17)	3.19
Vehicle0	846	18	(Van; remainder)	(23.64,76.36)	3.23
Ecoli1	336	7	(im; remainder)	(22.92,77.08)	3.36
New-thyroid2	215	5	(hypo; remainder)	(16.89,83.11)	4.92
New-thyroid1	215	5	(hyper; remainder)	(16.28,83.72)	5.14
Ecoli2	336	7	(pp; remainder)	(15.48,84.52)	5.46
Segment0	2308	19	(brickface; remainder)	(14.26,85.74)	6.01
Glass6	214	9	(headlamps; remainder)	(13.55,86.45)	6.38
Yeast3	1484	8	(me3; remainder)	(10.98,89.02)	8.11
Ecoli3	336	7	(imU; remainder)	(10.88,89.12)	8.19
Page-blocks0	5472	10	(remainder; text)	(10.23,89.77)	8.77
Ecoli034vs5	200	7	(p,imL,imU; om)	(10.00,90.00)	9.00
Yeast2vs4	514	8	(cyt; me2)	(9.92,90.08)	9.08
Ecoli067vs35	222	7	(cp,omL,pp; imL,om)	(9.91,90.09)	9.09
Ecoli0234vs5	202	7	(cp,imS,imL,imU; om)	(9.90,90.10)	9.10
Glass015vs2	172	9	(build-win-non_float-proc,tableware,build-win-float-proc; ve-win-float-proc)	(9.88,90.12)	9.12
Yeast0359vs78	506	8	(mit,me1,me3,erl; vac,pox)	(9.88,90.12)	9.12
Yeast02579vs368	1004	8	(mit,cyt,me3,vac,erl; me1,exc,pox)	(9.86,90.14)	9.14
Yeast0256vs3789	1004	8	(mit,cyt,me3,exc; me1,vac,pox,erl)	(9.86,90.14)	9.14
Ecoli046vs5	203	6	(cp,imU,omL; om)	(9.85,90.15)	9.15
Ecoli01vs235	244	7	(cp,im; imS,imL,om)	(9.83,90.17)	9.17
Ecoli0267vs35	224	7	(cp,imS,omL,pp; imL,om)	(9.82,90.18)	9.18
Glass04vs5	92	9	(build-win-float-proc,containers; tableware)	(9.78,90.22)	9.22
Ecoli0346vs5	205	7	(cp,imL,imU,omL; om)	(9.76,90.24)	9.25
Ecoli0347vs56	257	7	(cp,imL,imU,pp; om,omL)	(9.73,90.27)	9.28
Yeast05679vs4	528	8	(me2; mit,me3,exc,vac,erl)	(9.66,90.34)	9.35
Ecoli067vs5	220	6	(cp,omL,pp; om)	(9.09,90.91)	10.00
Vowel0	988	13	(hid; remainder)	(9.01,90.99)	10.10
Glass016vs2	192	9	(ve-win-float-proc; build-win-float-proc,build-win-non_float-proc,headlamps)	(8.89,91.11)	10.29
Glass2	214	9	(Ve-win-float-proc; remainder)	(8.78,91.22)	10.39
Ecoli0147vs2356	336	7	(cp,im,imU,pp; imS,imL,om,omL)	(8.63,91.37)	10.59
Led7digit02456789vs1	443	7	(0,2,4,5,6,7,8,9; 1)	(8.35,91.65)	10.97
Glass06vs5	108	9	(build-win-float-proc,headlamps; tableware)	(8.33,91.67)	11.00
Ecoli01vs5	240	6	(cp,im; om)	(8.33,91.67)	11.00
Glass0146vs2	205	9	(build-win-float-proc,containers,headlamps,build-win-non_float-proc; ve-win-float-proc)	(8.29,91.71)	11.06
Ecoli0147vs56	332	6	(cp,im,imU,pp; om,omL)	(7.53,92.47)	12.28
Cleveland0vs4	177	13	(0; 4)	(7.34,92.66)	12.62
Ecoli0146vs5	280	6	(cp,im,imU,omL; om)	(7.14,92.86)	13.00
Ecoli4	336	7	(om; remainder)	(6.74,93.26)	13.84
Yeast1vs7	459	8	(nuc; vac)	(6.72,93.28)	13.87
Shuttle0vs4	1829	9	(Rad Flow; Bypass)	(6.72,93.28)	13.87
Glass4	214	9	(containers; remainder)	(6.07,93.93)	15.47
Page-blocks13vs2	472	10	(graphic; horiz.line,picture)	(5.93,94.07)	15.85
Abalone9vs18	731	8	(18; 9)	(5.65,94.25)	16.68
Glass016vs5	184	9	(tableware; build-win-float-proc,build-win-non_float-proc,headlamps)	(4.89,95.11)	19.44
Shuttle2vs4	129	9	(Fpv Open; Bypass)	(4.65,95.35)	20.5
Yeast1458vs7	693	8	(vac; nuc,me2,me3,pox)	(4.33,95.67)	22.10
Glass5	214	9	(tableware; remainder)	(4.20,95.80)	22.81
Yeast2vs8	482	8	(pox; cyt)	(4.15,95.85)	23.10
Yeast4	1484	8	(me2; remainder)	(3.43,96.57)	28.41
Yeast1289vs7	947	8	(vac; nuc,cyt,pox,erl)	(3.17,96.83)	30.56
Yeast5	1484	8	(me1; remainder)	(2.96,97.04)	32.78
Ecoli0137vs26	281	7	(pp,imL; cp,im,imU,imS)	(2.49,97.51)	39.15
Yeast6	1484	8	(exc; remainder)	(2.49,97.51)	39.15
Abalone19	4174	8	(19; remainder)	(0.77,99.23)	128.87

5. Experimental study

In this section, we will perform an analysis to determine the performance of the different alternatives used for imbalanced classification. Our aim is to analyze three different issues:

1. The improvement obtained by preprocessing datasets and cost-sensitive learning over the original algorithm.
2. The possible differences between the rebalancing techniques versus cost-sensitive learning and in which cases.

Table 4
Parameter specification for the algorithms family employed in the experimentation.

Algorithm family	Parameters
C4.5	Pruned = true Confidence = 0.25 Minimum number of item-sets per leaf = 2
SVM	Kernel type = polynomial C = 100.0 Tolerance of termination criterion = 0.001 Degree (for kernel function) = 1 Gamma (for kernel function) = 0.01 coef0 (for kernel function) = 0.0 Use the shrinking heuristics = true
FH-GBML	Conjunction operator = product t-norm Rule weight = PCF (FH-GBML and FH-GBML + preprocessing) and PCF-SC (FH-GBML-CS) Fuzzy reasoning method = winning rule Number of fuzzy rules = $5 \cdot d$ (max. 50 rules) Number of rule sets = 200 Crossover probability = 0.9 Mutation probability = $1/d$ Number of replaced rules = all rules except the best-one (Pittsburgh-part, elitist approach) Number of rules/5 (GCCL-part) Total number of generations = 1.000 Do not care probability = 0.5 Probability of the application of the GCCL iteration = 0.5
k-NN	k = 3 Distance = Heterogeneous value difference metric (HVDM)

Table 5
Parameter specification for the preprocessing algorithms used in this study.

Preprocessing Algorithm	Parameters
SMOTE	kSMOTE = 5 Balancing = 1:1 distanceFunction = HVDM
SMOTE_ENN	kSMOTE = 5 kENN = 3 Balancing = 1:1 distanceFunction = HVDM

Table 6
Parameter specification for the wrapper routine.

Parameter	Value
Sample decrement	10%
Sample increment	100%
Increment min	5%
Look-ahead steps	2

3. Whether a hybrid methodology that combines a preprocessing approach and a cost-sensitive learning algorithm supposes a positive synergy and enables the achievement of more accurate results.

The study is divided into different paradigms to check whether the conclusions achieved for one paradigm can be extrapolated to the others.

5.1. Study of decision trees versions: C4.5

Table 7 shows the average results in training and test together with the corresponding standard deviation for the seven versions of the C4.5 algorithm used in the study: the base classifier, the base classifier used over the preprocessed datasets, the cost-sensitive version of the algorithm and the hybrid versions of it. We stress in **boldface** the best results achieved for the prediction ability of the different techniques. The complete table of results for all datasets is shown in the appendix of this work.

Table 7
Average table of results using the AUC measure for the C4.5 variety of algorithms.

Algorithm	AUC_{tr}	AUC_{tst}
C4.5	0.8774 ± 0.0392	0.7902 ± 0.0804
C4.5 SMOTE	0.9606 ± 0.0142	0.8324 ± 0.0728
C4.5 SENN	0.9471 ± 0.0154	**0.8390 ± 0.0772**
C4.5CS	0.9679 ± 0.0103	0.8294 ± 0.0758
C4.5 Wr_SMOTE	0.9679 ± 0.0103	0.8296 ± 0.0763
C4.5 Wr_US	0.9635 ± 0.0139	0.8245 ± 0.0760
C4.5 Wr_SENN	0.9083 ± 0.0377	0.8145 ± 0.0712

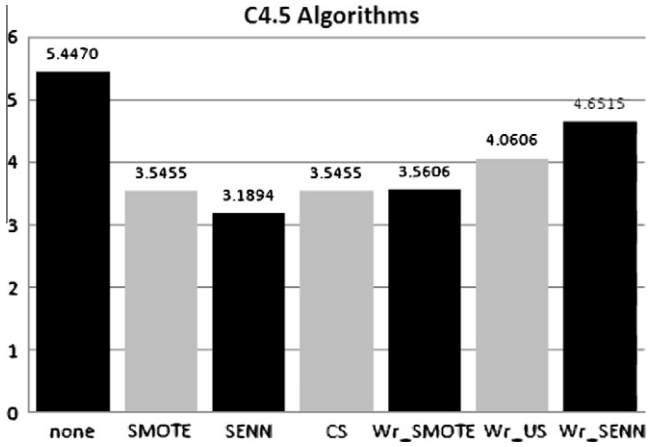


Fig. 6. Average rankings using the AUC measure for the C4.5 variety of algorithms.

From this table of results it can be observed that the highest average value corresponds to preprocessing approaches closely followed by the cost-sensitive learning approach and one version of the wrapper routine. This suggests the goodness of the preprocessing and cost-sensitive learning approaches.

In order to compare the results, a multiple comparison test is used to find the performance relationship between the different versions studied. The results of the statistical analysis of the C4.5 family are as follows. For the sake of a visual comparison, Fig. 6 shows the average ranking obtained through Friedman's test (Friedman, 1937) for these approaches. Under the AUC measure, the Iman–Davenport test detects significant differences among the algorithms, since the p-value returned (1.88673E−10) is lower than our α-value (0.05). The differences found are analyzed with a Shaffer test, shown in Table 8. In this table, a "+" symbol implies that the algorithm in the row is statistically better than the one in the column, whereas "−" implies the contrary; "=" means that the two algorithms compared have no significant differences. In brackets, the adjusted p-value associated to each comparison is shown.

Observing the results from Tables 7 and 8, we conclude that the standard C4.5 approach is outperformed by most of the methodologies that deal with imbalanced data. The base version is different from every other version except the hybrid version that uses only an oversampling step with SMOTE + ENN. Thus, we can state that the imbalanced classification approaches (preprocessing and cost-sensitive learning) improve the base classifier.

Comparing the results when applying preprocessing we can see that the performance of these methods is not statistically different for any of its versions. In addition, the performance of those preprocessing methods is also not different to the cost-sensitive

Table 8
Shaffer test for the C4.5 variety of algorithms using the AUC measure.

C4.5	None	SMOTE	SENN	CS	Wr_SMOTE	Wr_US	Wr_SENN
None	×	−(6.404E−6)	−(4.058E−8)	−(6.404E−6)	−(7.904E−6)	−(.00341)	=(.37846)
SMOTE	+(6.404E−6)	×	=(1.0)	=(1.0)	=(1.0)	=(1.0)	+(.04903)
SENN	+(4.058E−8)	=(1.0)	×	=(1.0)	=(1.0)	=(.22569)	+(.00152)
CS	+(6.404E−6)	=(1.0)	=(1.0)	×	=(1.0)	=(1.0)	+(.04903)
Wr_SMOTE	+(7.904E−6)	=(1.0)	=(1.0)	=(1.0)	×	=(1.0)	+(.04903)
Wr_US	+(.00341)	=(1.0)	=(.22569)	=(1.0)	=(1.0)	×	=(1.0)
Wr_SENN	=(.37846)	−(.04903)	−(.00152)	−(.04903)	−(.04903)	=(1.0)	×

Table 9
Average table of results using the AUC measure for the SVM variety of algorithms.

Algorithm	AUC_{tr}	AUC_{tst}
SVM	0.7563 ± 0.0198	0.7341 ± 0.0530
SVM SMOTE	0.8806 ± 0.0140	0.8514 ± 0.0568
SVM SENN	0.8826 ± 0.0146	**0.8517 ± 0.0557**
SVMCS	0.7869 ± 0.0281	0.7651 ± 0.0621
SVM Wr_SMOTE	0.6981 ± 0.0283	0.6820 ± 0.0521
SVM Wr_US	0.7077 ± 0.0315	0.6895 ± 0.0619
SVM Wr_SENN	0.7656 ± 0.0303	0.7461 ± 0.0662

learning version of C4.5. This means that in decision trees both preprocessing and cost-sensitive learning are good approaches to deal with the problem.

Focusing on the hybridization of cost-sensitive learning and preprocessing by using a wrapper routine, it can be seen that there are significant differences both between the different hybrid versions and with the other alternatives. The hybrid version that uses only an oversampling step with SMOTE + ENN is outperformed by all the other versions except the base version. The rest of the hybrid versions are not statistically different from the performance of usual approaches for imbalanced classification. Therefore, we cannot state that the hybridization in decision trees produces a positive synergy between the two techniques.

5.2. Study of support vector machines versions

In this part of the study, we follow the same scheme that was previously carried out. The average results are shown in Table 9 and, as in the former case, the complete table of results can be found in Appendix A of this work.

According to the results presented in Table 9, we may conclude that the preprocessing approaches perform better than the remaining proposals. We first check for significant differences using an Iman–Davenport test, which obtains a p-value (5.25259E−36) below our level of significance and near to zero. The associated statistical study is developed in Table 10, where we show the p-values computed by a Shaffer test with which we compare every SVM version using the AUC measure. In Fig. 7 the average ranking obtained through Friedman's test for these versions displayed, in which we can observe that the best rankings correspond to preprocessing approaches whereas worst rankings coincide with the hybrid approaches.

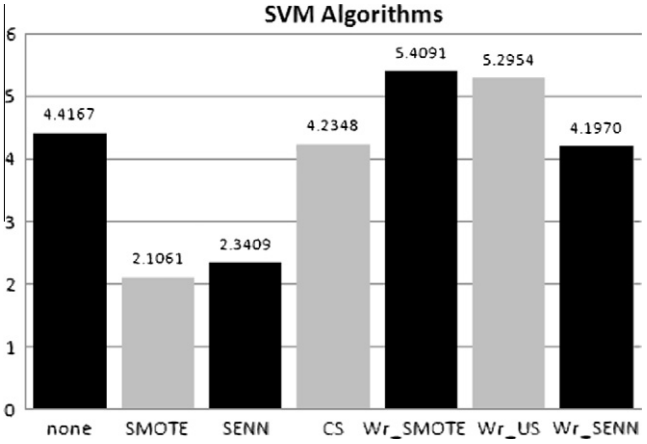


Fig. 7. Average rankings using the AUC measure for the SVM variety of algorithms.

Table 11
Average table of results using the AUC measure for the FH-GBML variety of algorithms.

Algorithm	AUC_{tr}	AUC_{tst}
FH-GBML	0.8352 ± 0.0226	0.7692 ± 0.0756
FH-GBML SMOTE	0.9181 ± 0.0130	0.8364 ± 0.0733
FH-GBML SENN	0.9127 ± 0.0131	0.8350 ± 0.0736
FH-GBMLCS	0.9328 ± 0.0076	**0.8373 ± 0.0773**
FH-GBML Wr_SMOTE	0.9330 ± 0.0075	0.8244 ± 0.0830
FH-GBML Wr_US	0.9304 ± 0.0095	0.8322 ± 0.0834
FH-GBML Wr_SENN	0.8866 ± 0.0306	0.8168 ± 0.0901

Table 10 shows that the original SVM is outperformed by the two preprocessing versions whereas there are not significant differences to the rest of versions. This means that the preprocessing approach improves the base classifier, however, the cost-sensitive learning proposal for SVMs is not competitive enough to be able to state that there are statistical differences. The hybridizations also cannot exceed the base classifier.

Comparing the results of preprocessing datasets we can see that the performance of these methods is not statistically different for

Table 10
Shaffer test for the SVM variety of algorithms using the AUC measure.

SVM	None	SMOTE	SENN	CS	Wr_SMOTE	Wr_US	Wr_SENN
None	×	−(1.364E−8)	−(4.749E−7)	=(1.0)	=(.05819)	=(.11667)	=(1.0)
SMOTE	+(1.364E−8)	×	=(1.0)	+(2.409E−7)	+(3.329E−17)	+(4.454E−16)	+(4.042E−7)
SENN	+(4.749E−7)	=(1.0)	×	+(6.167E−6)	+(6.421E−15)	+(7.094E−14)	+(9.585E−6)
CS	=(1.0)	−(2.409E−7)	−(6.167E−6)	×	+(.01792)	+(.03837)	=(1.0)
Wr_SMOTE	=(.05819)	−(3.329E−17)	−(6.421E−15)	−(.01792)	×	=(1.0)	−(.01394)
Wr_US	=(.11667)	−(4.454E−16)	−(7.094E−14)	−(.03837)	=(1.0)	×	−(.03139)
Wr_SENN	=(1.0)	−(4.042E−7)	−(9.585E−6)	=(1.0)	+(.01394)	+(.03139)	×

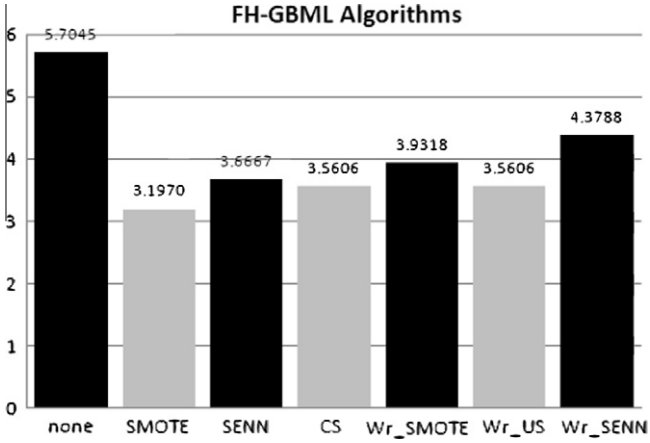


Fig. 8. Average rankings using the AUC measure for the FH-GBML variety of algorithms.

any of its versions. Nevertheless, there are significant differences between the preprocessing versions and the cost-sensitive learning version for SVMs. Furthermore, the preprocessing versions outperform statistically any other version.

If we just look at the hybridization of cost-sensitive learning and preprocessing by using a wrapper routine to check if the hybridization contributes to improve the cost-sensitive learning performance we find that there are significative differences between the different hybrid versions and between the other alternatives. The hybrid version that uses only an oversampling step with SMOTE + ENN outperforms the other hybrid versions whereas it has no significant differences with the cost-sensitive learning version. The rest of hybrids versions are not statistically different, however, they are also outperformed by the cost-sensitive version. In this paradigm, we cannot say that the hybridization produces a positive synergy between the two techniques because some of the hybrid versions are even outperformed by the cost-sensitive learning proposal.

5.3. Study of fuzzy rule based systems versions: FH-GBML

Table 11 shows the average results in training and test together with the corresponding standard deviation for the seven versions of the FH-GBML algorithm. The complete table of results for all datasets is also shown in Appendix A of this work together with the results of the previous experiments.

According to the average values shown in this table the best methods in this case are the preprocessing approaches and the cost-sensitive learning. To carry out the statistical study we first check for significant differences among the algorithms using an Iman–Davenport test. The p-value (8.20497E−12) is lower than our level of confidence $\alpha = 0.05$ and near to zero. Thus, we can conclude that significant differences do exist, proceeding with a Shaffer test. The ranks of the algorithms are presented in Fig. 8, and the results of the multiple comparison test performed on all algorithms are shown in Table 12.

Table 13
Average table of results using the AUC measure for the k-NN variety of algorithms.

Algorithm	AUC_{tr}	AUC_{tst}
3-NN	0.7697 ± 0.0555	0.7752 ± 0.0916
3-NN SMOTE	0.8880 ± 0.0495	0.8212 ± 0.0836
3-NN SENN	0.8743 ± 0.0434	0.8166 ± 0.0733
3-NNCS	0.8229 ± 0.0567	0.8295 ± 0.0854
3-NN Wr_SMOTE	0.8594 ± 0.0253	**0.8596 ± 0.0626**
3-NN Wr_US	0.8564 ± 0.0283	0.8561 ± 0.0655
3-NN Wr_SENN	0.8849 ± 0.0316	0.8509 ± 0.0664

At first glance we can check the tendency that we have seen in the previous studies: the base classifier is significantly different from other versions in the experimental study. Particularly, the base FH-GBML classifier is outperformed by the other versions, which means that the techniques used in imbalanced classification are useful and achieve better results than not performing special strategies to improve the results.

If we focus now on the performance of preprocessing methods we can observe that the oversampling versions are not statistically different. If we examine the preprocessing versions versus the cost-sensitive learning proposal we can see that they also do not differ statistically. With this information we can state that preprocessing and cost-sensitive learning are a good option to deal with the imbalanced classification problem.

Finally, we look at the hybridization versions from cost-sensitive learning and preprocessing. We find that between the different hybrid versions there are not statistical differences. If we compare the hybrid versions against the other versions of the study we can appreciate a difference between one of the hybrid versions and the cost-sensitive learning algorithm. Specifically, the cost-sensitive version surpasses the hybrid version that uses only an oversampling step with SMOTE + ENN. From this study, we cannot find a synergy in the hybridization.

5.4. Study of lazy learning versions: k-NN

Similar to the studies of other paradigms, we show in Table 13 the average results in training and test for the different versions of the study. We also refer the reader to the appendix for the complete table of results.

According to the average values shown in this table the best methods in this case seem to be the hybridizations of the preprocessing approaches with cost-sensitive learning. To carry out the statistical study we first check for significant differences among the algorithms using an Iman–Davenport test. The p-value (2.71648E−22) is lower than our level of confidence $\alpha = 0.05$ and near to zero. Thus, we can conclude that significant differences do exist, proceeding with a Shaffer test. The ranks of the algorithms are presented in Fig. 9, and the results of the multiple comparison test performed on all algorithms are shown in Table 14.

Table 12
Shaffer test for the FH-GBML variety of algorithms using the AUC measure.

FH-GBML	None	SMOTE	SENN	CS	Wr_SMOTE	Wr_US	Wr_SENN
None	×	−(5.439E−10)	−(1.078E−6)	−(2.379E−7)	−(4.128E−5)	−(2.379E−7)	−(.00676)
SMOTE	+(5.439E−10)	×	=(.64093)	=(1.0)	=(.41406)	=(1.0)	=(1.0)
SENN	+(1.078E−6)	=(.64093)	×	=(1.0)	=(1.0)	=(1.0)	=(.60824)
CS	+(2.379E−7)	=(1.0)	=(1.0)	×	=(1.0)	=(1.0)	+(.02511)
Wr_SMOTE	+(4.128E−5)	=(.41406)	=(1.0)	=(1.0)	×	=(1.0)	=(1.0)
Wr_US	+(2.379E−7)	=(1.0)	=(1.0)	=(1.0)	=(1.0)	×	=(.41406)
Wr_SENN	+(.00676)	=(1.0)	=(.60824)	−(.02511)	=(1.0)	=(.41406)	×

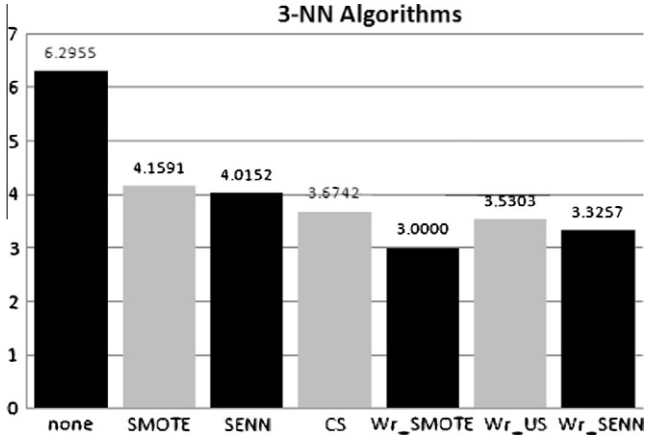


Fig. 9. Average rankings using the AUC measure for the k-NN variety of algorithms.

In this last part of experiments we confirm the tendency we pointed out after the previous statistical procedures: the base classifier is statistically different from the other versions used in the study. Using 3-NN the base classifier is outperformed by all the other algorithms in the analysis. This fact proves again that the approaches used in imbalanced classification are a need to solve these problems.

Moving to the comparison between preprocessing methods we can see that they do not differ statistically. If we broadened the comparison and we include the cost-sensitive learning proposal we still can see that there are no statistical differences. With these facts we can say that preprocessing and cost-sensitive learning are strong alternatives to solve the imbalanced classification problem.

The hybridization of cost-sensitive learning and preprocessing for 3-NN seems promising according to the average values. However, the Shaffer test does not indicate us the presence of statistical differences between the different hybrid versions. When we extend the comparison to the preprocessing and cost-sensitive learning versions we can find a difference between the base classifier combined with the SMOTE dataset and the hybrid version that uses only an oversampling step with SMOTE + ENN. Surprisingly, the difference in this case is in favor of the hybrid version. Due to these facts, for 3-NN we cannot say that there is no synergy between preprocessing and cost-sensitive learning; however, this improvement is so small that gets outshined by its bad results in the other paradigms.

5.5. General analysis on the suitability of preprocessing vs. cost-sensitive learning

As summary of the experimental study, and to unify the different conclusions extracted through the analysis of the results from the different selected paradigms, in this subsection we discuss the results we can discern attending to the three different issues we wanted to deal with: the first one devoted to demonstrate the goodness of both approaches for enhancing the performance of standard learning algorithms on this scenario, the second one for contrasting the behaviour of both preprocessing and cost-sensitive between them and the third part where a hybrid approach combining the two approaches is studied.

Before addressing the general conclusions we want to emphasize an idea about the generalization of these experiments: we cannot extrapolate the behaviour of a version from one paradigm to another. This fact has been critical in the hybrids models where a hybrid version was put at the same level of the base classifier in a paradigm whereas the same hybrid version outperformed a preprocessing approach in another paradigm.

Focusing now on the questions of the study, regarding the first issue, it is straightforward that classification performance is degraded in an imbalance scenario having a bias towards the majority class examples and that the use of the aforementioned techniques allow us to obtain a better discrimination of the examples of both classes resulting in an overall good classification for all concepts of the problem (positive and negative classes).

The second part of the study has reflected that the two employed solutions are quite similar between them and it was not possible to highlight one of them as the most adequate one for no one of the different type of algorithms (paradigms) selected for this study. Therefore, the question on which approach is preferable for addressing classification with imbalanced datasets is still unresolved.

Finally, the last approach differs from our expectations on a positive synergy. In most cases, the preliminary versions of hybridization techniques do not show a good behaviour in contrast to standard preprocessing and cost-sensitive learning. Nevertheless, some work on the combination of preprocessing and cost-sensitive learning can still be addressed with more specific methods that enhance the behaviour of these approaches.

6. Analyzing the limitations of both preprocessing and cost-sensitive learning in imbalanced classification. Open problems related to data intrinsic characteristics

According to the conclusions extracted in the previous section, we should focus on the nature of the problem itself in order to detect why both type of techniques obtain a comparable quality of solutions and how to address the imbalance problem in a more reasonable way. In this section we look at two data intrinsic characteristics issues, class overlapping and dataset shift, and their influence on imbalanced classification.

In Section 6.1 we will discuss some results about the influence of the imbalance ratio over the classification process and its relationship with the class overlap regions. Then, in Section 6.2 we will talk about the class overlapping problem and how it increases the difficulty to solve imbalanced classification problems. Finally, Section 6.3 will present the dataset shift problem and its relationship to imbalanced datasets classification.

6.1. On the influence of the imbalance ratio and its relationship with the class overlap regions

As we have stated previously, in real world machine learning applications, it has often been reported that the class imbalance

Table 14
Shaffer test for the k-NN variety of algorithms using the AUC measure.

3-NN	None	SMOTE	SENN	CS	Wr_SMOTE	Wr_US	Wr_SENN
None	×	−(2.142E−7)	−(2.260E−8)	−(5.690E−11)	−(3.981E−17)	−(3.679E−12)	−(5.711E−14)
SMOTE	+(2.142E−7)	×	=(1.0)	=(1.0)	−(.03081)	=(.80278)	=(.34698)
SENN	+(2.260E−8)	=(1.0)	×	=(1.0)	=(.09722)	=(1.0)	=(.80119)
CS	+(5.690E−11)	=(1.0)	=(1.0)	×	=(.94508)	=(1.0)	=(1.0)
Wr_SMOTE	+(3.981E−17)	+(.03081)	=(.09722)	=(.94508)	×	=(1.0)	=(1.0)
Wr_US	+(3.679E−12)	=(.80278)	=(1.0)	=(1.0)	=(1.0)	×	=(1.0)
Wr_SENN	+(5.711E−14)	=(.34698)	=(.80119)	=(1.0)	=(1.0)	=(1.0)	×

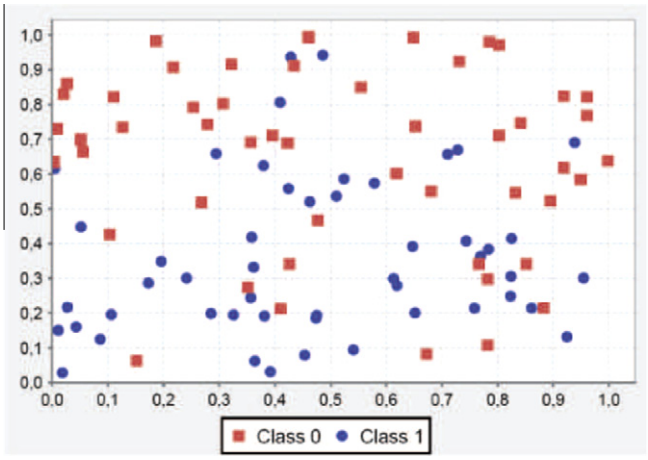


Fig. 10. $F1 = 0.6994$.

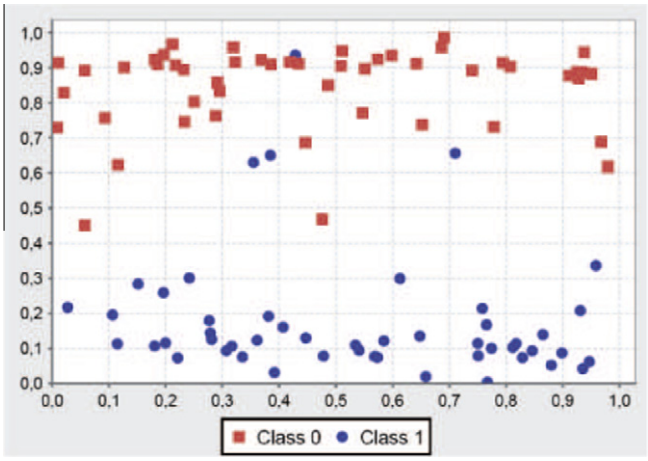


Fig. 11. $F1 = 9.69$.

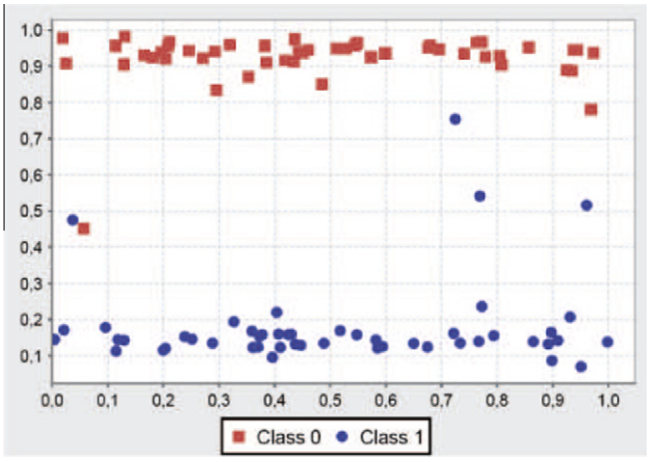


Fig. 12. $F1 = 26.16$.

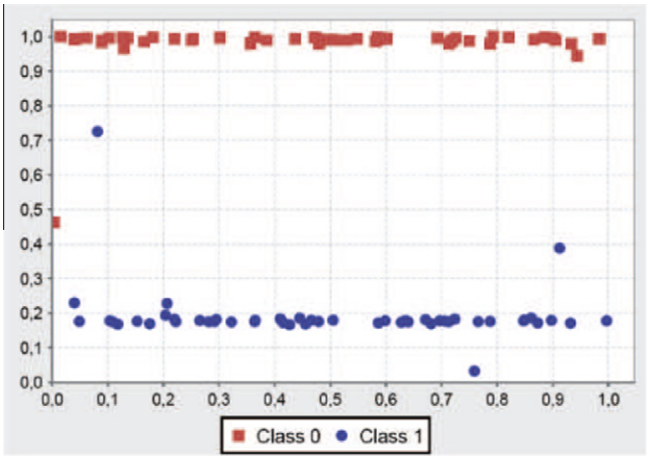


Fig. 13. $F1 = 48.65$.

hinders the performance of some standard classifiers. However, the relationship between class imbalance and learning algorithms is not clear yet, and a good understanding of how each one affects the other is lacking. Japkowicz and Stephen (2002) state that "Linearly separable domains are not sensitive to any amount of imbalance. As a matter of fact, as the degree of concept complexity increases, so does the system's sensitivity to imbalance". Thus, it does not seem fair to directly correlate class imbalance to the loss of performance of learning algorithms. Rather, it is quite possible that beyond class imbalances yield certain conditions that hamper classifiers induction.

As a direct result, there is a need to check whether class imbalances are truly to blame for the loss of performance of learning systems or whether the class imbalances are not a problem by themselves. Prati and Batista (2004) develop a study where they try to find an explanation for this performance decrease. Their experiments suggest that the problem is not solely caused by class imbalanced, but it is also related to the degree of data overlapping among the classes. They propose several experiments with synthetic datasets varying the IR and the overlap existing between the two classes. From them, it is deduced that it is not the class probabilities the main responsible for the hinder in the classification performance, but instead the degree of overlapping between the classes. This class overlapping may have a role even more important to concept induction than class imbalance. Thus, dealing with class imbalances will not always help classifiers performance improvement.

García et al. (2008) also develop a study focusing on the relationship between the IR and the overlap class regions. They studied the performance of several algorithms in different situations of imbalance and overlap focusing in the k-NN algorithm. For their study, they also use a set of synthetic datasets to check the relationship of these problems in several different situations. On the one hand, they try to find the relation when the IR in the overlap region is similar to the overall IR whereas on the other hand, they search for the relation when the IR in the overlap region is inverse to the overall one (the positive class is locally denser than the negative class in the overlap region). This first situation concludes that the increase in overlapping of a homogeneous imbalance affects more the (overall) minority class. Furthermore, the more local schemes tend to be better at classifying the minority class whereas models based on a more global learning are more robust at classifying the majority class. The second situation produces results where the accuracy on positive class is improved whereas negative class produces almost-stable accuracy curves. This example reveals that when the overlapped data is not balanced, the IR in overlapping can be more important than the overlapping size. In addition, classifiers based on more global learning attain greater TP rates whereas more local learning models obtain better TN rates than the former. This complementarity between global and local classifiers suggest a direction for future works on learning from imbalance data which will be discussed in Section 6.2.

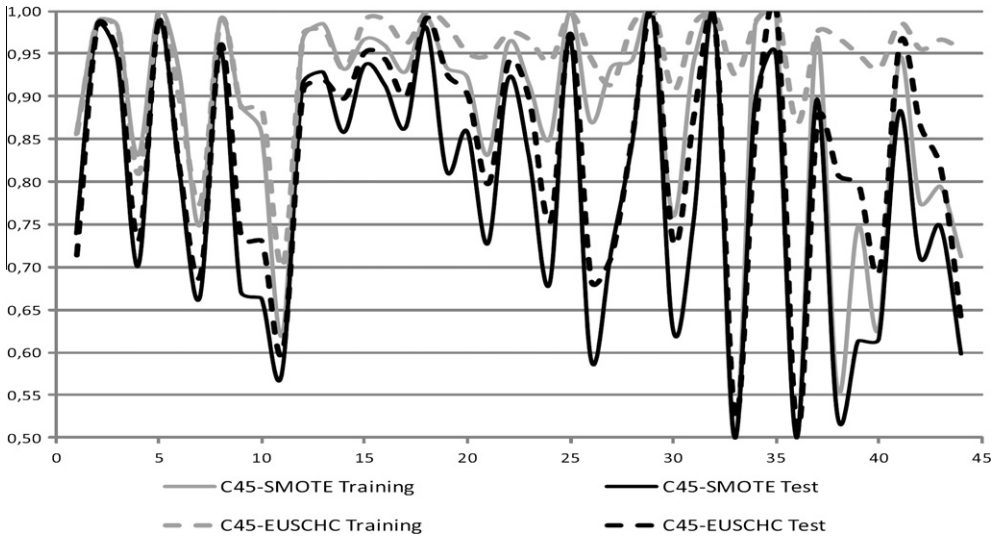


Fig. 14. Performance in training and test for the C4.5 decision tree with SMOTE sorted using the IR.

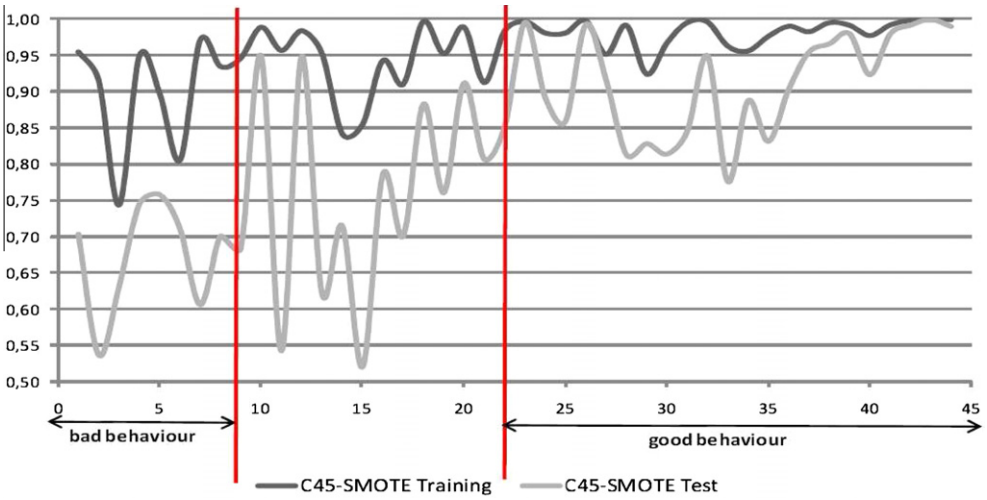


Fig. 15. Performance in training and test for the C4.5 decision tree with SMOTE sorted using the F1 data complexity metric.

Similarly, Denil and Trappenberg (2010) examine the effects of overlap and imbalance on the complexity of the learned model and demonstrate that overlap is a far more serious factor than imbalance in this respect. They demonstrate that these two problems acting in concert cause difficulties that are more severe than one would expect by examining their effects in isolation. In order to do so, they also use synthetic datasets for classifying with a SVM where they vary the IR, the overlap between classes and the IR and overlap jointly. Their results show that when the training set size is small, high levels of imbalance cause a dramatic drop in classifier performance, explained by the presence of small disjuncts. Overlapping classes cause a consistent drop in performance regardless of the size of the training set. However, with overlap and imbalance combined the classifier performance is degraded significantly beyond what the model predicts.

On the other hand, there exist recent works which have shown empirically with real world datasets that the quality of the results has not a clear relationship with the IR. Specifically, in Luengo et al. (2011) the authors try to characterize this datasets using complexity measures, which capture different aspects or sources of complexity which are considered complicated to the classification task. Specifically, they use the so called metric *F*1 or *maximum Fisher's discriminant ratio* (Ho & Basu, 2002) which measures the overlap of individual feature values. This data complexity metric, for one feature dimension, is defined as:

$$f = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \tag{13}$$

where μ_1, μ_2, σ_1 y σ_2 are the means and variances of the two classes, respectively, in that feature dimension. We compute f for each feature and take the maximum as measure F1. For a multidimensional problem, not all features have to contribute to class discrimination. The problem is easy as long as there exists one discriminating feature. Therefore, we can just take the maximum f over all feature dimensions in discussing class separability. Small values indicate that the classes have a high overlapping degree. Figs. 10–13 show illustrative examples of artificially generated data with two variables in the range [0.0; 1.0] and two classes as example, similar to those used in the studies from García et al. (2008) or Denil and Trappenberg (2010).

In Luengo et al. (2011) the authors depicted the performance of the different datasets ordered according to the imbalanced ratio and the F1 measure in order to search for some regions of

interesting good or bad behaviour. In Fig. 14 we can observe that the good and bad results of both learning methods with respect to the preprocessing are not related with the IR value, nor the improvements achieved with preprocessing steps. However, if the datasets are ordered using the F1 measure depicted in Fig. 15 both good and bad behavior intervals can be extracted, indicated by vertical lines. Therefore, the IR is not good enough to predict a classifier behavior and we need to focus on other problems to achieve better performance.

6.2. Addressing the significance of overlapping for imbalanced datasets

According to the studies previously presented, we observe the necessity to focus our efforts on the research for solutions in the imbalanced scenario towards the problem of overlapping between classes, without discarding in any case the issue of data distribution.

Our aim, given the current studies on the topic, is to address the overlapping problem integrating measures to deal with imbalance, opening many ways for future work. Therefore, following the approaches for imbalanced classification we can find several paths to improve the performance.

If we look at approaches at the **algorithm-level** we try to find algorithms that can show complementarity between global and local classifiers as suggested by García et al. (2008). A recently emerging solution to class imbalance is through the use of "information granulation". This high level procedure takes a less literal interpretation of data: instead of viewing a training sample as a set of example points, this type of classification tries to first establish higher level concepts via the construction of information granules. Kaburlasos (2007) propose a method that uses Fuzzy ART (Carpenter, Grossberg, & Rosen, 1991) to select a level of granularity. Based on these results, data is represented and a traditional learner is used. Fuzzy ART at its core is a clustering (unsupervised) system and this approach may be viewed as an additional feature transformation phase prior to classification. Chen et al. (2008) apply a similar framework, although k-means clustering is used to determine information granules instead of Fuzzy Art.

Regarding FRBCSs, Fernández, del Jesus, and Herrera (2009) proposed to make use of a Hierarchical FRBCS, which consists in the application of a thicker granularity in order to generate the initial rule base, and to reinforce those problem subspaces that are specially difficult by means of the application of rules with a higher granularity. Also, in Gama (2004) the author uses a framework of decision trees which allows to, for those leaves which have difficulties to discriminate between examples of different classes, to apply a strong classifier (for example an SVM or any other technique) in order to obtain a better separability in this specific area of the problem, rather than just using a standard heuristic.

Therefore, a very positive approach at the algorithm-level could consist in working with different granular levels, in a way that more general submodels of knowledge could cover the largest part of the problem space, whereas in more difficult areas, that is, boundary zones with a high degree of overlapping, we could use more specific discrimination functions in different paradigms of learning algorithms.

If we now turn a look at preprocessing approaches at the **data-level** we have in mind a double objective: try to find algorithms that can balance the data distribution whereas trying to avoid overlap as much as possible.

In oversampling techniques, and specially for the SMOTE algorithm, the problem of over generalization is largely attributed to the way in which it creates synthetic samples. Specifically, SMOTE generates the same number of synthetic data samples for each original minority example and does so without consideration to neighboring examples, which increases the occurrence of overlapping between classes (Wang & Japkowicz, 2004). To this end, various adaptive sampling methods have been proposed to overcome this limitation; some representative works include the Borderline-SMOTE (Han, Wang, & Mao, 2005), Adaptive Synthetic Sampling (He, Bai, Garcia, & Li, 2008) and the Safe-Level-SMOTE (Bunkhumpornpat, Sinapiromsaran, & Lursinsap, 2009) algorithms. In Fig. 16 we can observe the generation of new instances using an over-sampling algorithm. It defines three kind of instances according to its neighbors: "Safe" instances that can be used to generate synthetic samples, "Danger" instances that can be used to generate synthetic samples but can introduce overlap between the two classes and "Noise" instances that should not be used to generate instances as they are considered wrongly labelled instances.

Also, some combination of preprocessing of instances with data cleaning techniques could lead to diminish the overlapping that is introduced from sampling methods. Some representative work in this area includes the one-sided selection method Kubat and Matwin (1997), the condensed nearest neighbor rule and Tomek Links integration method Batista et al. (2004), the neighborhood cleaning rule Laurikkala (2001) based on the edited nearest neighbor (ENN) rule which removes examples that differ from two of its three nearest neighbors, and the integrations of SMOTE with ENN and SMOTE with Tomek links Batista et al. (2004) (Fig. 17).

In this manner, applying new ways of informed preprocessing techniques in order to identify and weight significant samples and discard noisy examples in the boundary areas could be an interesting topic for future work for both relaxing overlapping

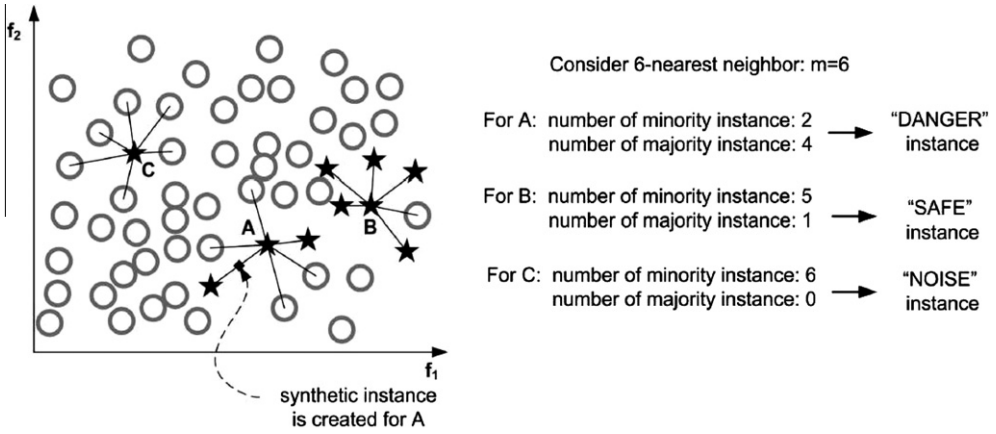


Fig. 16. Data creation based on Borderline instance.

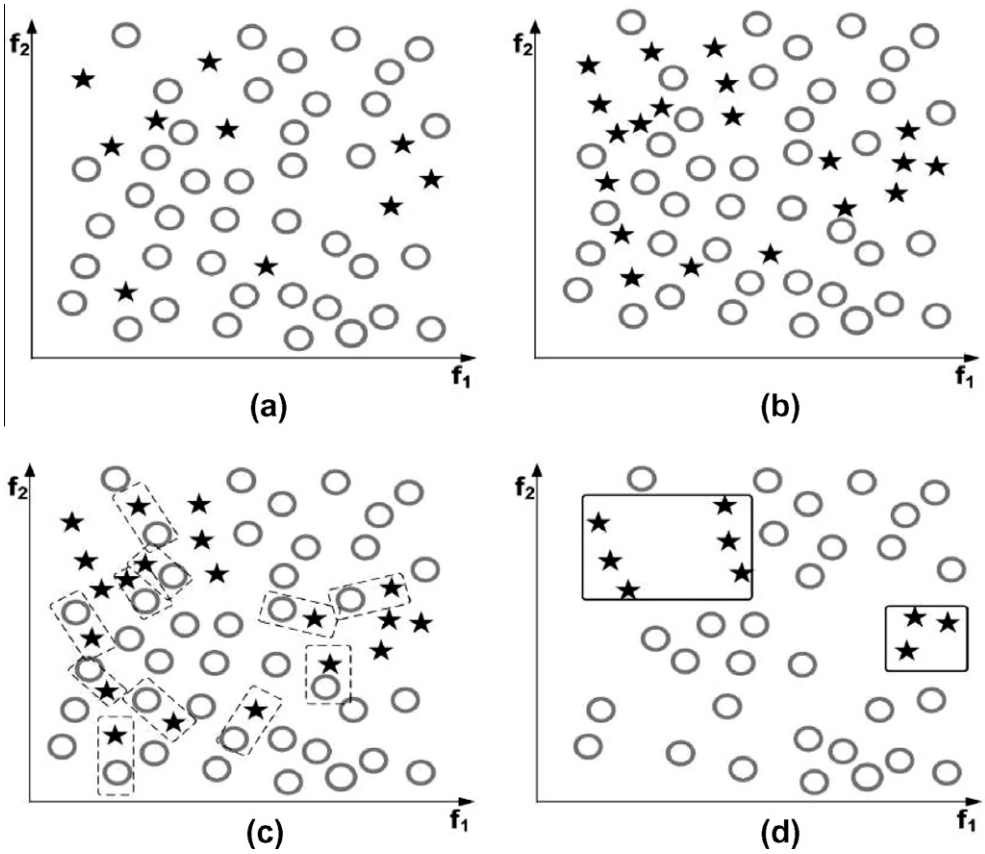


Fig. 17. (a) Original dataset distribution, (b) post-SMOTE dataset, (c) the identified Tomek links, and (d) the dataset after removing Tomek links.

and rebalancing the data distribution for avoiding the bias towards the majority class examples during the learning stage.

Still in the preprocessing approach, Martín-Félez and Mollineda (2010) propose an approach that combines preprocessing and feature selection (specifically in this order), in a way that preprocessing deals with data distribution and small disjuncts ("IR part") and feature selection somehow reduces the degree of overlapping ("F1 part"). In a more general way, the idea behind this approach consists on overcoming different sources of data complexity such as class overlap, irrelevant and redundant features, noisy samples, class imbalance, low ratios of the sample size to dimensionality and so on using different approaches used to solve each complexity.

Also, Villar, Fernández, Sánchez, and Herrera (2009, 2010) use a FRBCS that performs an *a priori* learning of the data base to obtain the optimal number of variables and granularity level for the fuzzy partitions in an imbalance scenario. The authors combine preprocessing (SMOTE in this case) with the former technique with very good results in performance (in contrast with C4.5) with a reduction of about the 65% of the variables for high imbalanced problems.

In summary, in order to reduce the original overlapping of a problem, we may apply a feature selection process in order to remove those instances which do not introduce any relevant information but makes hard to obtain discrimination functions for a given dataset.

6.3. Dataset shift in imbalanced classification

The problem of dataset shift (Alaiz-Rodríguez & Japkowicz, 2008; Shimodaira, 2000; Quiñonero Candela, Sugiyama, Schwaighofer, & Lawrence, 2009) is defined as the case where training and test data follow different distributions. This is a common problem that can affect all kind of classification problems, and it often appears due to sample selection bias issues. A mild degree of dataset shift is present in most real-world problems, but general classifiers are often capable of handling it without a severe performance loss.

However, the dataset shift issue is specially relevant when dealing with imbalanced classification, because in highly imbalanced domains, the minority class is particularly sensitive to singular classification errors, due to the typically low number of examples it presents (Moreno-Torres & Herrera, 2010). In the most extreme cases, a single misclassified example of the minority class can create a significant drop in performance.

Fig. 18 presents an example of dataset shift in imbalanced classification for clarity. Note how, in the test set, some of the minority class examples are in an area where there was none in the training set, creating a situation that is likely to produce low classifier performance.

Since dataset shift is a highly relevant issue in imbalanced classification, it is easy to see why it would be an interesting perspective to focus on in future research regarding the topic.

There are two different potential approaches in the study of the effect and solution of dataset shift in imbalanced domains. The first one focuses on intrinsic dataset shift, that is, the data of interest includes some degree of shift that is producing a relevant drop in performance. In this case, we need to:

- Develop techniques to discover and measure the presence of dataset shift following the suggestions made in (Cieslak & Chawla, 2009; Wang et al., 2003; Yang, Wu, & Zhu, 2008); but adapting them to focus on the minority class. To do so, either a partially labeled test set will be needed (in order to properly identify the minority class examples in the test set), or a new strategy will have to be developed.

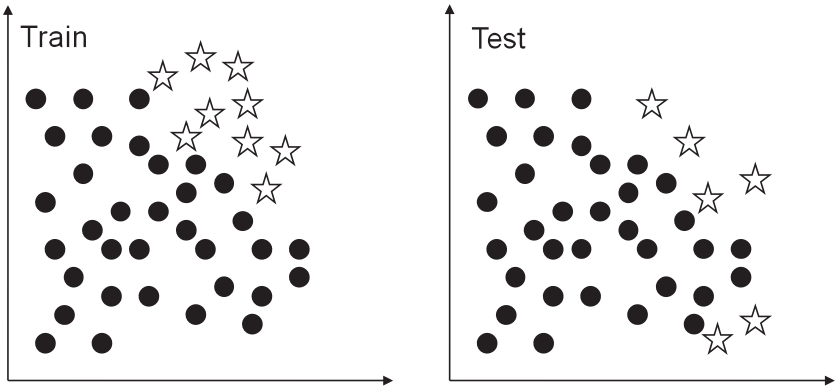


Fig. 18. Example of the impact of dataset shift in imbalanced domains.

- Design algorithms that are capable of working under dataset shift conditions. These could be either preprocessing techniques (Moreno-Torres, Llorà, Goldberg, & Bhargava, in press) or algorithms that are designed to have the capability to adapt and deal with dataset shift without the need for a preprocessing step (Alaiz-Rodríguez, Guerrero-Curieses, & Cid-Sueiro, 2009; Bickel, Brückner, & Scheffer, 2009; Globerson, Teo, Smola, & Roweis, 2009; Gretton et al., 2009; Sugiyama, Krauledat, & Müller, 2007). In both cases, we are not aware of any proposals in the literature that focus on the problem of imbalanced classification in the presence of dataset shift.

The second branch in terms of dataset shift in imbalanced classification is related to induced dataset shift. Most current state of the art research is validated through stratified cross-validation techniques, which are another potential source of shift in the machine learning process. A more suitable validation technique needs to be developed in order to avoid introducing dataset shift issues artificially.

7. Concluding remarks

In this work we have analyzed the preprocessing performance in the framework of imbalanced datasets against other approaches in this problem such as cost-sensitive learning. We have considered two oversampling methods: SMOTE and SMOTE + ENN, a cost-sensitive version and a hybrid approach that tries to integrate both approaches together.

We have observed that the approaches used to address the imbalanced problem improve the overall performance in all the paradigms used in the study, which was the expected behaviour.

The comparison between preprocessing techniques against cost-sensitive learning hints that there are no differences among the different preprocessing techniques. The statistical study carried out let us say that both preprocessing and cost-sensitive learning are good and equivalent approaches to address the imbalance problem.

The preliminary versions of hybridization techniques are truly competitive with the standard methodologies only in some cases, which determines more work needs to be done in addressing this approach.

Finally, we develop a discussion about how to go above preprocessing and cost-sensitive learning limits. We try to analyze the problem according to the results and we focus on the open problems to improve the algorithms. Specifically, we have emphasized that the IR is important but there are still other issues like the class overlapping and dataset shift problems that arise in some cases and can prove detrimental in terms of classification performance. Since overcoming these problems is the key to the improvement of the algorithms' performance, future work should be oriented to analyze the existing overlap to create accurate algorithms that can improve in imbalanced classification and to use dataset shift repairing techniques to fill the gap between data distributions.

Acknowledgments

This work has been supported by the Spanish Ministry of Education and Science under Project TIN2008-06681-C06-01. V. López and J.G. Moreno-Torres hold FPU scholarships from Spanish Ministry of Education.

Appendix A. Detailed results for the experimental study

In this appendix we present the complete results tables for all the algorithms used in this work. Thus, the reader can observe the full training and test results, in order to compare the performance of each approach. In Table A.15 we show the

Table A.15
Complete table of results using the AUC measure for the C4.5 variety of algorithms.

C4.5	None		SMOTE		SENN		CS		Wr_SMOTE		Wr_US		Wr_SENN	
Dataset	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst
Glass1	0.8561	**0.7399**	0.9234	0.7368	0.8690	0.6921	0.9069	0.7160	0.9069	0.7160	0.8831	0.6682	0.8595	0.7367
Ecoli0vs1	0.9870	**0.9832**	0.9926	0.9729	0.9870	**0.9832**	0.9870	**0.9832**	0.9870	**0.9832**	0.9800	**0.9832**	0.9870	**0.9832**
Wisconsin	0.9840	0.9454	0.9826	0.9532	0.9776	0.9576	0.9780	**0.9636**	0.9780	**0.9636**	0.9768	0.9555	0.9755	0.9524
Pima	0.8317	0.7012	0.8179	0.7245	0.8012	**0.7403**	0.8571	0.7125	0.8571	0.7125	0.8621	0.7311	0.8203	0.7077
Iris0	1.0000	**0.9900**	1.0000	**0.9900**	1.0000	**0.9900**	1.0000	**0.9900**	1.0000	**0.9900**	1.0000	**0.9900**	1.0000	**0.9900**
Glass0	0.9306	0.8167	0.9459	0.7752	0.8897	0.7994	0.9205	**0.8212**	0.9205	**0.8212**	0.9100	0.8042	0.8636	0.7999
Yeast1	0.7494	0.6642	0.8085	**0.7090**	0.7829	0.6954	0.7855	0.6779	0.7855	0.6779	0.7806	0.6767	0.8023	0.6945
Vehicle1	0.8898	0.6717	0.9503	0.7301	0.8817	**0.7542**	0.9362	0.7013	0.9362	0.7013	0.9276	0.7130	0.8173	0.6719

Table A.15 (*continued*)

C4.5	None		SMOTE		SENN		CS		Wr_SMOTE		Wr_US		Wr_SENN	
Dataset	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst
Vehicle2	0.9905	**0.9561**	0.9905	0.9498	0.9848	0.9419	0.9866	0.9434	0.9866	0.9434	0.9850	0.9450	0.9811	0.9419
Vehicle3	0.8588	0.6637	0.9508	0.7282	0.8914	**0.7409**	0.9221	0.7283	0.9221	0.7283	0.9275	0.7010	0.8569	0.6791
Haberman	0.6204	0.5757	0.7124	**0.6163**	0.7383	0.5884	0.6380	0.5752	0.6380	0.5752	0.5879	0.5476	0.6417	0.5396
Glass0123vs456	0.9671	0.9155	0.9868	0.9232	0.9635	**0.9240**	0.9806	0.8777	0.9806	0.8777	0.9783	0.8931	0.9513	0.9108
Vehicle0	0.9863	0.9296	0.9878	0.9188	0.9724	0.9072	0.9861	0.9289	0.9861	0.9289	0.9799	**0.9373**	0.9752	0.9178
Ecoli1	0.9329	0.8586	0.9507	0.9105	0.9335	0.8926	0.9457	**0.9114**	0.9457	**0.9114**	0.9394	0.9017	0.9368	0.9065
New-thyroid2	0.9679	0.9373	0.9922	0.9659	0.9817	0.9774	0.9903	**0.9802**	0.9903	**0.9802**	0.9868	0.9437	0.9744	0.9063
New-thyroid1	0.9607	0.9143	0.9879	0.9631	0.9944	**0.9889**	0.9903	0.9746	0.9903	0.9746	0.9882	0.9746	0.9774	0.9405
Ecoli2	0.9297	0.8641	0.9738	0.8811	0.9716	**0.8976**	0.9594	0.8905	0.9594	0.8905	0.9515	0.8641	0.9473	0.8580
Segment0	0.9932	0.9826	0.9986	**0.9927**	0.9989	0.9916	0.9988	0.9919	0.9988	0.9919	0.9967	0.9894	0.9940	0.9876
Glass6	0.9347	0.8132	0.9872	0.8842	0.9851	0.9203	0.9865	0.8896	0.9865	0.8896	0.9878	0.8923	0.9369	**0.9365**
Yeast3	0.9237	0.8597	0.9607	0.8905	0.9617	**0.9230**	0.9784	0.9117	0.9784	0.9117	0.9796	0.9096	0.9587	0.9176
Ecoli3	0.8320	0.7280	0.9671	0.8123	0.9371	**0.8705**	0.9585	0.8326	0.9585	0.8326	0.9605	0.8452	0.9133	0.8694
Page-blocks0	0.9637	0.9221	0.9848	**0.9504**	0.9797	0.9427	0.9903	0.9458	0.9903	0.9458	0.9894	0.9435	0.9614	0.9284
Ecoli034vs5	0.9188	0.8389	0.9854	0.9000	0.9764	0.8806	0.9938	**0.9250**	0.9938	**0.9250**	0.9896	0.8972	0.9694	0.9111
Yeast2vs4	0.9158	0.8307	0.9814	0.8588	0.9746	**0.9042**	0.9797	0.8866	0.9797	0.8866	0.9768	0.8955	0.9323	0.8291
Ecoli067vs35	0.8789	0.8250	0.9781	0.8500	0.9775	0.8125	0.9875	0.8825	0.9875	0.8825	0.9869	0.8775	0.9201	**0.8875**
Ecoli0234vs5	0.9313	0.8307	0.9897	**0.8974**	0.9828	0.8947	0.9966	0.8334	0.9966	0.8334	0.9835	0.7946	0.9730	0.8835
Glass015vs2	0.8910	0.5011	0.9766	0.6772	0.9083	**0.7957**	0.9790	0.6003	0.9790	0.6003	0.9758	0.5938	0.8727	0.5508
Yeast0359vs78	0.7028	0.5868	0.9490	**0.7047**	0.9217	0.7024	0.9715	0.6765	0.9715	0.6765	0.9556	0.6721	0.8362	0.6641
Yeast02579vs368	0.8809	0.8432	0.9767	**0.9143**	0.9576	0.9138	0.9874	0.8996	0.9874	0.8996	0.9855	0.8896	0.9533	0.9102
Yeast0256vs3789	0.7563	0.6606	0.9330	**0.7951**	0.9179	0.7817	0.9743	0.7846	0.9743	0.7846	0.9435	0.7403	0.8906	0.7648
Ecoli046vs5	0.9368	0.8418	0.9870	0.8701	0.9836	**0.8869**	0.9911	0.8310	0.9911	0.8310	0.9884	0.8174	0.9543	0.7978
Ecoli01vs235	0.9097	0.7136	0.9656	**0.8377**	0.9650	0.8332	0.9739	0.7641	0.9739	0.7641	0.9727	0.7664	0.9263	0.7532
Ecoli0267vs35	0.8788	0.7752	0.9796	0.8155	0.9827	0.8179	0.9889	0.8527	0.9889	0.8527	0.9852	**0.8653**	0.9067	0.8577
Glass04vs5	0.9940	**0.9941**	0.9910	0.9816	0.9910	0.9754	0.9940	**0.9941**	0.9940	**0.9941**	0.9940	**0.9941**	0.9940	**0.9941**
Ecoli0346vs5	0.9118	0.8615	0.9892	**0.8980**	0.9885	**0.8980**	0.9905	0.8507	0.9905	0.8507	0.9905	0.8534	0.9579	0.7730
Ecoli0347vs56	0.8600	0.7757	0.9778	**0.8568**	0.9568	0.8546	0.9892	0.7586	0.9898	0.7764	0.9806	0.7985	0.9384	0.8100
Yeast05679vs4	0.8508	0.6802	0.9526	0.7602	0.9199	0.7802	0.9741	0.7243	0.9741	0.7243	0.9691	0.7480	0.9134	**0.7804**
Ecoli067vs5	0.9363	0.7675	0.9875	0.8475	0.9744	0.8450	0.9888	**0.8825**	0.9888	**0.8825**	0.9869	0.8775	0.9081	0.8600
Vowel0	0.9999	**0.9706**	0.9971	0.9505	0.9943	0.9455	0.9925	0.9422	0.9925	0.9422	0.9928	0.9311	0.9928	0.9322
Glass016vs2	0.8710	0.5938	0.9716	0.6062	0.9375	**0.6388**	0.9829	0.6155	0.9829	0.6155	0.9807	0.5793	0.8529	0.5788
Glass2	0.9350	0.7194	0.9700	0.6390	0.9280	**0.7457**	0.9734	0.6416	0.9734	0.6416	0.9639	0.6715	0.8669	0.6501
Ecoli0147vs2356	0.8578	0.8051	0.9789	0.8277	0.9565	0.8228	0.9882	0.8772	0.9882	0.8772	0.9866	**0.8788**	0.9112	0.7673
Led7digit02456789vs1	0.9022	0.8788	0.9225	**0.8908**	0.9249	0.8379	0.9203	0.8436	0.9203	0.8436	0.9178	0.8387	0.9042	0.8616
Glass06vs5	0.9950	**0.9950**	0.9912	0.9147	0.9912	0.9647	0.9950	**0.9950**	0.9950	**0.9950**	0.9637	0.9579	0.9950	**0.9950**
Ecoli01vs5	0.9114	0.8159	0.9886	0.7977	0.9830	0.8250	0.9778	0.8182	0.9778	0.8182	0.9858	**0.8318**	0.9392	0.8136
Glass0146vs2	0.7879	0.6616	0.9676	**0.7842**	0.9042	0.7095	0.9847	0.6797	0.9847	0.6797	0.9708	0.6421	0.7930	0.6102
Ecoli0147vs56	0.8842	0.8318	0.9798	**0.8592**	0.9610	0.8424	0.9756	0.8539	0.9756	0.8539	0.9813	0.8371	0.9468	0.7774
Cleveland0vs4	0.8648	0.6878	0.9939	**0.7908**	0.9816	0.7605	0.9886	0.6893	0.9906	0.6823	0.9914	0.6885	0.9086	0.6795
Ecoli0146vs5	0.9178	0.7885	0.9870	**0.8981**	0.9851	**0.8981**	0.9808	0.8385	0.9808	0.8385	0.9837	0.8135	0.9572	0.8212
Ecoli4	0.9430	0.8437	0.9703	0.7794	0.9827	**0.9044**	0.9680	0.8636	0.9680	0.8636	0.9684	0.8636	0.9505	0.8386
Yeast1vs7	0.7608	0.6275	0.9351	0.7003	0.9097	**0.7371**	0.9741	0.6139	0.9741	0.6139	0.9671	0.6794	0.8530	0.6627
Shuttle0vs4	1.0000	0.9997	0.9999	0.9997	0.9999	0.9997	1.0000	0.9997	1.0000	0.9997	1.0000	**1.0000**	1.0000	**1.0000**
Glass4	0.9403	0.7542	0.9901	**0.8867**	0.9670	0.8650	0.9104	0.8431	0.9104	0.8431	0.9340	0.8298	0.8861	0.7831
Page-blocks13vs2	0.9989	0.9978	0.9975	0.9955	0.9975	0.9910	0.9989	**0.9789**	0.9989	**0.9789**	0.9977	0.9978	0.9791	0.9498
Abalone9vs18	0.6907	0.5859	0.9142	0.6283	0.9058	**0.7193**	0.9864	0.6655	0.9864	0.6655	0.9849	0.6369	0.8515	0.7150
Glass016vs5	0.9843	0.8943	0.9921	0.8129	0.9864	0.8629	0.9914	**0.9886**	0.9914	**0.9886**	0.9914	**0.9886**	0.9914	**0.9886**
Shuttle2vs4	1.0000	0.9500	0.9990	0.9917	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**
Yeast1458vs7	0.5000	0.5000	0.9158	0.5367	0.8770	0.5563	0.9640	0.5540	0.9640	0.5540	0.9625	**0.5464**	0.7788	0.4943
Glass5	0.9702	0.8976	0.9921	0.8805	0.9705	0.7756	0.9976	0.9427	0.9976	0.9427	0.9872	**0.9854**	0.8624	0.8439
Yeast2vs8	0.5563	0.5250	0.9071	0.8338	0.8960	0.8197	0.9927	**0.8652**	0.9927	0.8652	0.9913	0.8359	0.8541	0.7978
Yeast4	0.7482	0.6135	0.9071	0.7121	0.9007	0.7257	0.9722	0.7222	0.9722	0.7222	0.9700	0.6999	0.8872	**0.7400**
Yeast1289vs7	0.6290	0.6156	0.9465	0.6832	0.9414	0.6332	0.9752	0.6769	0.9752	0.6769	0.9748	**0.6973**	0.7073	0.6107
Yeast5	0.9453	0.8833	0.9777	0.9337	0.9820	0.9406	0.9929	0.9330	0.9929	0.9330	0.9928	0.9326	0.9743	**0.9434**
Ecoli0137vs26	0.7953	0.7481	0.9678	0.8136	0.9660	0.8136	0.9804	0.8281	0.9804	0.8281	0.9594	0.7954	0.8907	**0.8445**
Yeast6	0.7762	0.7115	0.9326	**0.8294**	0.9314	0.8270	0.9883	0.8082	0.9883	0.8082	0.9864	0.8099	0.8165	0.7311
Abalone19	0.5000	0.5000	0.8550	0.5205	0.8890	0.5166	0.9839	**0.5701**	0.9839	**0.5701**	0.9835	0.5543	0.6211	0.5231
Average	0.8774	0.7902	0.9606	0.8324	0.9471	**0.8390**	0.9679	0.8294	0.9679	0.8296	0.9635	0.8245	0.9083	0.8145

Table A.16
Complete table of results using the AUC measure for the SVM variety of algorithms.

SVM	None		SMOTE		SENN		CS		Wr_SMOTE		Wr_US		Wr_SENN	
Dataset	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst
Glass1	0.5155	0.4963	0.6613	0.6179	0.6780	0.6391	0.6624	**0.6264**	0.5000	0.5000	0.5097	0.5000	0.6229	0.5682
Ecoli0vs1	0.9675	0.9671	0.9844	**0.9796**	0.9811	0.9770	0.9675	0.9671	0.9844	**0.9796**	0.9810	0.9731	0.9828	**0.9796**
Wisconsin	0.9728	0.9666	0.9770	0.9727	0.9794	0.9691	0.9724	0.9719	0.9653	0.9552	0.9726	0.9626	0.9777	**0.9737**
Pima	0.7334	0.7194	0.7523	0.7348	0.7520	0.7300	0.7378	0.7289	0.6985	0.6916	0.6960	0.7116	0.7452	**0.7449**
Iris0	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**

(*continued on next page*)

Table A.16 (*continued*)

SVM	None		SMOTE		SENN		CS		Wr_SMOTE		Wr_US		Wr_SENN	
Dataset	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst
Glass0	0.7070	0.6914	0.7716	0.7377	0.7755	0.7243	0.5215	0.5074	0.7778	0.7517	0.7778	0.7517	0.7411	**0.7520**
Yeast1	0.5771	0.5732	0.7108	**0.7100**	0.7096	0.7067	0.6675	0.6749	0.5000	0.5000	0.5012	0.5000	0.6750	0.6897
Vehicle1	0.7510	0.7202	0.8001	0.7742	0.8184	**0.8055**	0.7930	0.7546	0.6401	0.6180	0.6147	0.6076	0.7125	0.6882
Vehicle2	0.9693	0.9527	0.9722	**0.9601**	0.9711	0.9578	0.9734	0.9571	0.9223	0.9068	0.9371	0.9070	0.9023	0.8891
Vehicle3	0.7290	0.7134	0.7805	0.7613	0.8101	0.7881	0.8072	**0.7904**	0.4789	0.4871	0.5612	0.5753	0.6339	0.6306
Haberman	0.5223	0.5036	0.6287	**0.6344**	0.6621	0.6332	0.5225	0.5382	0.5000	0.5000	0.5000	0.5000	0.5217	0.4996
Glass0123vs456	0.9151	0.9043	0.9351	**0.9050**	0.9426	0.8987	0.8572	0.8445	0.8572	0.8445	0.8672	0.8445	0.9425	0.8987
Vehicle0	0.9780	0.9490	0.9778	0.9632	0.9778	0.9611	0.9781	0.9493	0.9798	0.9620	0.9805	**0.9653**	0.9610	0.9470
Ecoli1	0.8331	0.8192	0.9082	**0.9062**	0.9006	0.9024	0.9084	**0.9062**	0.6430	0.6367	0.6523	0.6535	0.8776	0.8659
New-thyroid2	0.9972	0.9829	0.9965	**0.9917**	0.9917	0.9889	0.9972	0.9829	0.9750	0.9687	0.9802	0.9603	0.9680	0.9659
New-thyroid1	0.9972	0.9829	0.9965	**0.9944**	0.9944	0.9861	0.9943	0.9687	0.9786	0.9516	0.9901	0.9829	0.9701	0.9603
Ecoli2	0.7675	0.7351	0.9073	**0.9067**	0.9065	0.9050	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.8916	0.8805
Segment0	0.9954	0.9927	0.9987	0.9955	0.9985	**0.9965**	0.9990	**0.9965**	0.9947	0.9932	0.9946	0.9932	0.9944	0.9922
Glass6	0.9379	**0.9198**	0.9497	0.9063	0.9554	0.9009	0.8882	0.8725	0.8882	0.8725	0.8964	0.8919	0.9281	0.9032
Yeast3	0.6305	0.6299	0.9056	0.8917	0.9114	**0.9061**	0.9057	0.8951	0.5000	0.5000	0.5000	0.5000	0.5200	0.5154
Ecoli3	0.5000	0.5000	0.9037	**0.8984**	0.8964	0.8818	0.8222	0.7925	0.5000	0.5000	0.5855	0.5614	0.7267	0.6976
Page-blocks0	0.8287	0.8218	0.9251	0.9258	0.9292	**0.9273**	0.9248	0.9254	0.5001	0.5004	0.4976	0.4769	0.5738	0.5828
Ecoli034vs5	0.9153	0.8611	0.9271	**0.8889**	0.9250	0.8861	0.8750	0.8639	0.8750	0.8639	0.8847	0.8556	0.8972	**0.8889**
Yeast2vs4	0.6691	0.6691	0.9090	**0.8896**	0.9084	0.8885	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.7408	0.7360
Ecoli067vs35	0.8999	**0.8525**	0.9276	0.8325	0.9239	0.8350	0.8363	0.8025	0.7807	0.7050	0.8468	0.8300	0.8733	0.8275
Ecoli0234vs5	0.9229	0.8667	0.9302	**0.8892**	0.9205	**0.8892**	0.8813	0.8417	0.8813	0.8417	0.8834	0.8140	0.9292	0.8696
Glass015vs2	0.5000	0.5000	0.5943	0.5094	0.5961	**0.5191**	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Yeast0359vs78	0.6067	0.6067	0.7476	**0.7451**	0.7522	0.7450	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.6067	0.6067
Yeast02579vs368	0.8090	0.8006	0.9137	0.9013	0.9143	**0.9069**	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.8890	0.8981
Yeast0256vs3789	0.5524	0.5486	0.8102	0.7940	0.8098	**0.8018**	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.8034	0.8000
Ecoli046vs5	0.9028	0.8696	0.9213	**0.8869**	0.9130	**0.8869**	0.8875	0.8696	0.8875	0.8696	0.8806	0.8669	0.8966	0.8642
Ecoli01vs235	0.8863	0.8359	0.9393	0.8505	0.9420	**0.8550**	0.8429	0.7805	0.8429	0.7805	0.8796	0.8582	0.9029	0.7959
Ecoli0267vs35	0.8899	0.8526	0.9162	0.8255	0.9156	**0.8530**	0.8346	0.7851	0.8346	0.7851	0.8288	0.8251	0.8717	0.8079
Glass04vs5	0.8893	0.8500	0.9638	**0.9566**	0.9638	0.9507	0.8893	0.9000	0.8893	0.9000	0.8983	0.9129	0.8893	0.9000
Ecoli0346vs5	0.9035	0.8696	0.9191	0.8926	0.9287	0.8926	0.8688	0.8946	0.8688	0.8946	0.8743	**0.8973**	0.9279	0.8088
Ecoli0347vs56	0.9123	0.8935	0.9219	**0.9082**	0.9224	0.9061	0.8550	0.8135	0.8500	0.8135	0.8545	0.8135	0.9191	0.8848
Yeast05679vs4	0.5000	0.5000	0.8016	**0.8075**	0.7977	0.7875	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.7371	0.7135
Ecoli067vs5	0.9094	0.8425	0.9213	0.8475	0.9238	0.8075	0.8500	0.7450	0.8500	0.7450	0.8775	0.8325	0.9013	**0.9125**
Vowel0	0.9096	0.8950	0.9793	**0.9622**	0.9795	**0.9622**	0.8655	0.8461	0.9432	0.9244	0.9420	0.9172	0.9477	0.9489
Glass016vs2	0.5000	0.5000	0.6462	**0.5336**	0.6520	0.5267	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Glass2	0.5000	0.5000	0.6883	0.6155	0.6852	**0.6905**	0.7051	0.5953	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Ecoli0147vs2356	0.8635	0.8434	0.8973	**0.8828**	0.9060	0.8727	0.7801	0.7267	0.7801	0.7267	0.7882	0.7101	0.8885	0.8568
Led7digit02456789vs1	0.9051	**0.8901**	0.8981	0.8851	0.8850	0.8891	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.9027	0.8272
Glass06vs5	0.7071	0.6500	0.9520	**0.9437**	0.9520	0.9437	0.6929	0.6500	0.6929	0.6500	0.8262	0.6245	0.6929	0.6500
Ecoli01vs5	0.9273	0.8364	0.9648	0.8364	0.9608	0.8364	0.8813	0.7909	0.8813	0.7909	0.8864	0.7909	0.9403	**0.8864**
Glass0146vs2	0.5000	0.5000	0.6631	0.6121	0.6729	**0.6310**	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Ecoli0147vs56	0.9080	**0.8719**	0.9181	0.8612	0.9205	0.8546	0.8400	0.7967	0.8400	0.7967	0.8742	0.8335	0.8984	0.8519
Cleveland0vs4	0.9403	0.7483	0.9619	0.8785	0.9627	**0.9149**	0.9318	0.7483	0.9318	0.7483	0.9503	0.7483	0.8966	0.8014
Ecoli0146vs5	0.8798	0.8635	0.9269	**0.8904**	0.9404	0.8808	0.8438	0.7923	0.8438	0.7923	0.8620	0.8154	0.8865	0.8654
Ecoli4	0.5875	0.5750	0.9743	0.9200	0.9739	0.9200	0.9834	**0.9529**	0.5000	0.5000	0.5000	0.5000	0.6313	0.6000
Yeast1vs7	0.5000	0.5000	0.7746	**0.7861**	0.7664	0.7741	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Shuttle0vs4	1.0000	**1.0000**	1.0000	0.9960	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	0.9960	1.0000	**1.0000**
Glass4	0.6157	0.5592	0.9546	**0.9576**	0.9621	0.9101	0.9615	0.9126	0.6064	0.5617	0.5964	0.5592	0.7529	0.6733
Page-blocks13vs2	0.8896	0.8332	0.9654	0.9561	0.9654	**0.9640**	0.8513	0.8566	0.6777	0.7757	0.6654	0.6325	0.7104	0.6738
Abalone9vs18	0.5029	0.5000	0.8161	0.8127	0.8257	0.8128	0.8352	**0.8740**	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Glass016vs5	0.5839	0.4971	0.9536	0.9429	0.9521	**0.9457**	0.5554	0.5000	0.5554	0.5000	0.6346	0.5886	0.5825	0.5471
Shuttle2vs4	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	0.9990	0.9960	1.0000	**1.0000**
Yeast1458vs7	0.5000	0.5000	0.6926	**0.6373**	0.7032	0.6266	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Glass5	0.5554	0.5000	0.9518	0.9512	0.9488	0.9415	0.9713	**0.9732**	0.5554	0.5000	0.5554	0.5500	0.5143	0.5000
Yeast2vs8	0.7739	0.7739	0.8201	0.7663	0.8183	0.7642	0.8223	0.7664	0.5500	0.5739	0.5500	0.5739	0.7739	**0.7739**
Yeast4	0.5000	0.5000	0.8571	0.8241	0.8560	**0.8258**	0.8604	0.8155	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Yeast1289vs7	0.5000	0.5000	0.7401	**0.7194**	0.7455	0.7077	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Yeast5	0.5000	0.5000	0.9641	0.9653	0.9642	0.9628	0.9648	**0.9656**	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Ecoli0137vs26	0.8733	**0.8500**	0.9571	0.7990	0.9521	0.8044	0.8733	**0.8500**	0.8733	**0.8500**	0.8720	0.8481	0.8553	0.8463
Yeast6	0.5000	0.5000	0.8886	0.8730	0.8867	0.8696	0.8807	**0.8758**	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Abalone19	0.5000	0.5000	0.8039	**0.7930**	0.8150	0.7873	0.8170	0.7615	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
Average	0.7563	0.7341	0.8806	0.8514	0.8826	**0.8517**	0.7869	0.7651	0.6981	0.6820	0.7077	0.6895	0.7656	0.7461

Table A.17
Complete table of results using the AUC measure for the FH-GBML variety of algorithms.

FH-GBML	None		SMOTE		SENN		CS		Wr_SMOTE		Wr_US		Wr_SENN	
Dataset	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst
Glass1	0.8103	0.7199	0.8194	0.7313	0.8220	0.7331	0.8270	0.7414	0.8263	0.6852	0.8278	0.6964	0.8192	**0.7540**
Ecoli0vs1	0.9958	0.9762	0.9926	0.9627	0.9837	0.9532	0.9942	**0.9765**	0.9959	0.9729	0.9928	0.9550	0.9878	0.9698
isconsin	0.9818	0.9620	0.9811	0.9638	0.9785	0.9720	0.9828	**0.9780**	0.9841	0.9704	0.9829	0.9704	0.9739	0.9507

Table A.17 (*continued*)

FH-GBML	None		SMOTE		SENN		CS		Wr_SMOTE		Wr_US		Wr_SENN	
Dataset	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst
Pima	0.7410	0.6980	0.7684	**0.7381**	0.7494	0.7061	0.7772	0.7274	0.7770	0.7235	0.7776	0.7304	0.7619	0.7321
Iris0	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	0.9950	0.9713	0.9850	1.0000	**1.0000**
Glass0	0.8246	0.7524	0.8515	0.7542	0.8325	0.7901	0.8500	0.7709	0.8565	0.8036	0.8542	0.7738	0.8342	**0.8043**
Yeast1	0.6717	0.6611	0.7310	0.7004	0.7211	0.7044	0.7353	0.7016	0.7351	0.7115	0.7346	**0.7234**	0.7272	0.6965
Vehicle1	0.6642	0.6164	0.7655	0.7106	0.7469	0.7049	0.7615	0.6982	0.7655	0.7126	0.7549	**0.7202**	0.7284	0.6902
Vehicle2	0.8257	0.8204	0.8917	0.8718	0.8812	0.8697	0.8900	0.8732	0.8932	**0.8768**	0.8859	0.8704	0.8916	0.8582
Vehicle3	0.6454	0.6121	0.7520	0.7128	0.7501	**0.7275**	0.7500	0.6942	0.7485	0.6929	0.7493	0.6966	0.7339	0.6748
Haberman	0.6284	0.5130	0.7080	0.6136	0.6851	0.6067	0.7498	0.6061	0.7547	0.6132	0.7506	**0.6141**	0.6345	0.5163
Glass0123vs456	0.9651	0.8888	0.9722	0.9307	0.9704	**0.9430**	0.9796	0.9154	0.9774	0.8474	0.9803	0.9082	0.9617	0.8954
Vehicle0	0.8618	0.8348	0.9062	0.8938	0.9007	0.8697	0.8997	0.8878	0.9194	**0.9053**	0.9131	0.9050	0.8988	0.8837
Ecoli1	0.9083	0.8480	0.9276	**0.8763**	0.9174	0.8704	0.9346	0.8659	0.9303	0.8732	0.9297	0.8731	0.9198	0.8557
New-thyroid2	0.9893	0.9546	0.9986	**0.9802**	0.9931	0.9774	1.0000	0.9516	1.0000	0.9544	0.9979	0.9659	0.9579	0.8944
New-thyroid1	0.9982	**0.9931**	1.0000	0.9516	0.9944	0.9917	1.0000	0.9659	1.0000	0.9659	0.9958	0.9405	0.9608	0.9488
Ecoli2	0.9296	0.8550	0.9538	0.8861	0.9471	**0.9369**	0.9569	0.8974	0.9564	0.9044	0.9536	0.8943	0.9428	0.9343
Segment0	0.9724	0.9709	0.9837	0.9772	0.9829	0.9741	0.9891	0.9806	0.9827	**0.9828**	0.9855	0.9802	0.9840	0.9736
Glass6	0.9656	0.9032	0.9772	0.8827	0.9743	0.8298	0.9854	0.8384	0.9949	0.8605	0.9899	0.8771	0.9509	**0.9252**
Yeast3	0.8673	0.8321	0.9432	0.9293	0.9362	0.9165	0.9447	0.9076	0.9419	0.9212	0.9424	**0.9298**	0.9341	0.9089
Ecoli3	0.8240	0.7674	0.9405	0.8847	0.9443	0.8787	0.9516	**0.8864**	0.9554	0.8502	0.9524	0.8772	0.9222	0.8283
Page-blocks0	0.8170	0.8116	0.9012	0.8938	0.8939	0.8983	0.9028	0.8944	0.9003	0.9017	0.8996	**0.9023**	0.8927	0.8868
Ecoli034vs5	0.9743	0.8569	0.9865	0.8944	0.9865	0.8444	0.9997	**0.9125**	1.0000	0.8236	0.9979	0.8861	0.9597	0.8972
Yeast2vs4	0.8859	0.8328	0.9442	0.9073	0.9504	0.8972	0.9626	0.8931	0.9610	0.9056	0.9606	**0.9196**	0.9019	0.8809
Ecoli067vs35	0.9324	0.8575	0.9458	0.8125	0.9539	**0.8750**	0.9828	0.8188	0.9831	0.8075	0.9863	0.8375	0.9036	0.8350
Ecoli0234vs5	0.9688	0.8890	0.9856	0.8572	0.9769	0.8434	0.9993	0.8059	0.9979	0.8696	0.9903	0.8227	0.9501	**0.9306**
Glass015vs2	0.5886	0.4887	0.8709	0.6008	0.8576	**0.7204**	0.9246	0.6481	0.9267	0.6191	0.9141	0.7167	0.7967	0.6013
Yeast0359vs78	0.6100	0.5889	0.7995	0.7226	0.7977	0.7351	0.8204	**0.7573**	0.8234	0.7030	0.8262	0.6879	0.7895	0.7004
Yeast02579vs368	0.8998	0.8619	0.9248	0.9099	0.9232	0.8938	0.9330	0.9001	0.9325	0.8982	0.9311	**0.9071**	0.9270	0.9029
Yeast0256vs3789	0.7259	0.6911	0.8283	0.7851	0.8252	0.7942	0.8374	0.7945	0.8388	0.7818	0.8359	**0.7970**	0.8226	0.7778
Ecoli046vs5	0.9688	0.8973	0.9877	0.8326	0.9829	0.8061	0.9986	**0.9669**	0.9973	0.8142	0.9963	0.8669	0.9682	0.9337
Ecoli01vs235	0.9407	0.7882	0.9693	0.8075	0.9625	**0.8482**	0.9781	0.7955	0.9804	0.8409	0.9794	0.8320	0.9276	0.7900
Ecoli0267vs35	0.9314	**0.8551**	0.9599	0.8331	0.9479	0.7991	0.9864	0.8315	0.9842	0.8103	0.9855	0.8303	0.9326	0.8216
Glass04vs5	1.0000	0.8441	0.9868	**0.9673**	0.9925	0.8574	1.0000	0.9199	1.0000	0.9375	0.9895	0.7195	0.9687	0.8188
Ecoli0346vs5	0.9556	0.7946	0.9823	0.8331	0.9872	0.9142	0.9986	0.8919	0.9990	0.8669	0.9926	0.9061	0.9627	**0.9223**
Ecoli0347vs56	0.9339	0.8357	0.9663	0.8600	0.9608	0.8525	0.9855	0.8320	0.9847	0.8737	0.9844	0.8731	0.9423	**0.8792**
Yeast05679vs4	0.7084	0.6514	0.8559	**0.8064**	0.8456	0.7312	0.8690	0.7703	0.8665	0.7842	0.8693	0.7832	0.8476	0.7782
Ecoli067vs5	0.9375	0.8613	0.9600	0.8338	0.9656	0.8750	0.9903	0.8613	0.9897	0.8863	0.9869	0.8150	0.9050	**0.9125**
Vowel0	0.8924	0.8256	0.9661	**0.9561**	0.9565	0.9135	0.9663	0.9394	0.9630	0.9352	0.9563	0.9352	0.9521	0.9466
Glass016vs2	0.5727	0.5233	0.8671	0.6343	0.8498	**0.6895**	0.9046	0.6636	0.8973	0.5976	0.8912	0.5860	0.8092	0.5400
Glass2	0.5659	0.4885	0.8603	0.6771	0.8210	0.5991	0.8972	0.7098	0.9050	**0.8172**	0.8957	0.5978	0.7961	0.6106
Ecoli0147vs2356	0.8934	0.7936	0.9467	0.8508	0.9489	0.8457	0.9651	0.8622	0.9624	0.8077	0.9607	**0.8792**	0.8995	0.8043
Led7digit02456789vs1	0.9069	**0.8938**	0.9235	0.8839	0.9039	0.8900	0.9440	0.8745	0.9454	0.8741	0.9459	0.8666	0.9079	0.8823
Glass06vs5	1.0000	0.8925	0.9859	0.9320	0.9862	0.8925	1.0000	0.9100	1.0000	0.8747	0.9975	0.8950	0.9756	**0.9374**
Ecoli01vs5	0.9750	0.8648	0.9892	**0.8989**	0.9835	0.8864	0.9994	0.8432	1.0000	0.8875	0.9966	0.8886	0.9543	0.8693
Glass0146vs2	0.5368	0.4961	0.8510	0.7064	0.8352	0.6345	0.9111	**0.7618**	0.8996	0.6367	0.8947	0.6756	0.8079	0.7020
Ecoli0147vs56	0.9296	0.8667	0.9669	0.8045	0.9648	0.8605	0.9862	**0.8955**	0.9888	0.8388	0.9866	0.8596	0.9561	0.8820
Cleveland0vs4	0.9219	0.6939	0.9431	0.7520	0.9317	0.7056	0.9832	0.6861	0.9798	0.6348	0.9829	**0.7876**	0.9519	0.7541
Ecoli0146vs5	0.9495	0.7913	0.9786	**0.9202**	0.9856	0.8750	0.9990	0.8529	0.9983	0.7808	0.9962	0.9000	0.9418	0.8231
Ecoli4	0.9563	0.8703	0.9876	0.9302	0.9858	0.9294	0.9972	**0.9421**	0.9968	0.8873	0.9972	0.8905	0.9484	0.8913
Yeast1vs7	0.6786	0.5358	0.8396	0.7191	0.8543	0.6424	0.8673	**0.7389**	0.8773	0.7026	0.8724	0.6655	0.8012	0.6882
Shuttle0vs4	1.0000	0.9960	1.0000	0.9980	1.0000	**1.0000**	1.0000	0.9920	1.0000	0.9958	1.0000	**1.0000**	1.0000	0.9958
Glass4	0.9021	0.6479	0.9775	0.8867	0.9657	**0.9613**	0.9969	0.8746	0.9963	0.7505	0.9957	0.8684	0.9259	0.6868
Page-blocks13vs2	0.9375	0.9272	0.9866	0.9515	0.9882	0.9459	0.9958	0.9749	0.9949	**0.9787**	0.9959	0.9498	0.9532	0.9142
Abalone9vs18	0.6085	0.5912	0.7917	0.7165	0.7979	0.7376	0.8440	0.7737	0.8308	0.7774	0.8346	0.7797	0.7972	**0.7948**
Glass016vs5	0.9107	0.8136	0.9752	**0.8993**	0.9768	0.8921	0.9993	0.8193	1.0000	0.8443	0.9975	0.8300	0.9486	0.8964
Shuttle2vs4	1.0000	0.9500	1.0000	0.9940	1.0000	0.9877	1.0000	**1.0000**	1.0000	0.8500	1.0000	0.9500	0.9200	0.8500
Yeast1458vs7	0.5333	0.4985	0.7761	0.6287	0.7620	**0.6597**	0.8021	0.6319	0.7925	0.6370	0.7955	0.6237	0.7385	0.5822
Glass5	0.8797	0.8201	0.9899	0.7671	0.9848	0.7970	0.9988	0.8841	0.9994	0.7427	0.9976	**0.9201**	0.9636	0.8165
Yeast2vs8	0.8125	0.7478	0.8723	0.7442	0.8555	0.7226	0.8877	0.7411	0.8916	0.7839	0.8892	**0.8180**	0.8196	0.7076
Yeast4	0.5659	0.5167	0.8806	0.8137	0.8785	0.7947	0.8945	**0.8222**	0.8962	0.8027	0.8898	0.8214	0.8261	0.7394
Yeast1289vs7	0.6250	0.5820	0.8096	**0.7238**	0.7943	0.7175	0.8425	0.6393	0.8369	0.7076	0.8457	0.6441	0.6868	0.5299
Yeast5	0.7206	0.6783	0.9735	0.9469	0.9796	**0.9778**	0.9885	0.9740	0.9875	0.9314	0.9861	0.9396	0.9575	0.8958
Ecoli0137vs26	0.8767	0.7472	0.9824	0.8236	0.9820	0.8208	0.9991	0.7891	0.9989	0.8363	0.9966	**0.8445**	0.8544	0.7982
Yeast6	0.6243	0.6270	0.9204	0.8646	0.9215	0.8591	0.9296	0.8426	0.9317	**0.8713**	0.9302	0.8300	0.8716	0.8302
Abalone19	0.5000	0.5000	0.8322	0.6708	0.8250	**0.7297**	0.8387	0.6627	0.8493	0.6816	0.8321	0.6914	0.6293	0.5726
Average	0.8352	0.7692	0.9181	0.8364	0.9127	0.8350	0.9328	**0.8373**	0.9330	0.8244	0.9304	0.8322	0.8866	0.8168

Table A.18
Complete table of results using the AUC measure for the k-NN variety of algorithms.

3-NN	None		SMOTE		SENN		CS		Wr_SMOTE		Wr_US		Wr_SENN	
Dataset	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst
Glass1	0.7583	0.7460	0.8273	0.7805	0.8398	0.7761	0.7583	0.7460	0.7583	0.7460	0.7567	0.7350	0.8593	**0.8147**
Ecoli0vs1	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.9691	0.9600	0.9690	**0.9766**	0.9707	0.9533

(*continued on next page*)

Table A.18 (*continued*)

3-NN	None		SMOTE		SENN		CS		Wr_SMOTE		Wr_US		Wr_SENN	
Dataset	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst	Tr	Tst
Wisconsin	0.9636	0.9658	0.0214	0.9698	0.0209	**0.9729**	0.9636	0.9658	0.9636	0.9658	0.9641	0.9658	0.9647	0.9658
Pima	0.6686	0.6703	0.7479	0.6865	0.7682	0.7099	0.6686	0.6703	0.6686	0.6703	0.6696	0.6711	0.7986	**0.7297**
Iris0	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**	0.9988	**1.0000**	1.0000	**1.0000**
Glass0	0.8144	0.8027	0.8184	0.8185	0.8299	**0.8361**	0.7884	0.7771	0.7884	0.7771	0.7529	0.7670	0.8025	0.7809
Yeast1	0.6559	0.6539	0.7864	0.6772	0.7734	0.7071	0.6740	0.6858	0.6740	0.6858	0.6745	0.6841	0.7772	**0.7190**
Vehicle1	0.6741	0.6314	0.8454	0.6985	0.8230	**0.7752**	0.7665	0.7476	0.7665	0.7476	0.7664	0.7474	0.7781	0.7472
Vehicle2	0.9743	**0.9736**	0.9753	0.9692	0.9690	0.9620	0.9578	0.9541	0.9578	0.9541	0.8942	0.8960	0.9635	0.9519
Vehicle3	0.6395	0.6529	0.8539	0.7085	0.8291	**0.7636**	0.7365	0.7355	0.7365	0.7355	0.7367	0.7355	0.7416	0.7474
Haberman	0.5463	0.5310	0.6955	0.5633	0.6906	0.5767	0.6167	0.6510	0.6167	0.6510	0.6100	**0.6516**	0.6546	0.5729
Glass0123vs456	0.8859	0.8888	0.9709	0.9164	0.9620	0.9334	0.9424	0.9331	0.9424	0.9331	0.9338	**0.9399**	0.9407	0.9199
Vehicle0	0.9446	0.9379	0.9548	0.9471	0.9493	0.9415	0.9473	0.9461	0.9473	0.9461	0.9371	0.9363	0.9535	**0.9479**
Ecoli1	0.7693	0.7636	0.8484	0.8085	0.8345	0.8089	0.8019	0.8036	0.8789	0.8749	0.8721	0.8730	0.9165	**0.9065**
New-thyroid2	0.9508	0.9373	0.9889	0.9889	0.9875	0.9861	0.9831	**0.9917**	0.9831	**0.9917**	0.9854	0.9833	0.9688	0.9516
New-thyroid1	0.9401	0.9659	0.9917	0.9889	0.9889	0.9861	0.9831	**0.9917**	0.9831	**0.9917**	0.9818	0.9806	0.9816	0.9631
Ecoli2	0.8253	0.8302	0.8674	0.8382	0.8622	0.8276	0.8307	0.8276	0.9102	0.9154	0.9082	0.9066	0.9396	**0.9294**
Segment0	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.9970	**0.9980**	0.9941	0.9937	0.9977	0.9962
Glass6	0.9147	0.9140	0.9824	**0.9419**	0.9770	0.9338	0.9366	**0.9419**	0.9366	**0.9419**	0.9305	0.9365	0.9286	**0.9419**
Yeast3	0.8231	0.8171	0.9541	0.8681	0.9470	0.8634	0.8827	0.8777	0.8827	0.8777	0.8859	0.8803	0.9300	**0.8824**
Ecoli3	0.6798	0.6598	0.8514	0.7283	0.8443	0.7772	0.7766	0.7502	0.8432	0.8209	0.8428	0.8478	0.8932	**0.8863**
Page-blocks0	0.9040	0.9075	0.9796	0.9326	0.9748	0.9316	0.9409	**0.9370**	0.9409	0.9370	0.9409	**0.9370**	0.9530	0.9193
Ecoli034vs5	0.8306	0.8222	0.8813	0.8222	0.8813	0.8222	0.8556	0.8361	0.9389	**0.9333**	0.9389	0.9222	0.9535	0.9167
Yeast2vs4	0.7485	0.7368	0.8573	0.8073	0.8554	0.8073	0.7903	0.7938	0.8680	0.8771	0.8677	0.8771	0.9128	**0.8803**
Ecoli067vs35	0.7109	0.7625	0.8531	0.8200	0.8525	0.8150	0.7724	0.8550	0.8563	**0.8900**	0.8623	0.8800	0.8627	0.8275
Ecoli0234vs5	0.8125	0.8500	0.8746	0.8530	0.8746	0.8530	0.8328	0.8612	0.9294	0.9308	0.9329	0.9280	0.9446	**0.9336**
Glass015vs2	0.5943	0.5788	0.8637	0.6750	0.8637	0.6935	0.7036	**0.7097**	0.7036	**0.7097**	0.7073	0.6685	0.7225	0.6067
Yeast0359vs78	0.6522	0.6468	0.8736	**0.7247**	0.8733	0.7203	0.6762	0.6923	0.6762	0.6923	0.6852	0.6979	0.8005	0.6977
Yeast02579vs368	0.8845	0.8834	0.9507	0.9024	0.9506	0.9013	0.8970	0.8988	0.8970	0.8988	0.8852	0.8922	0.9269	**0.9082**
Yeast0256vs3789	0.7580	0.7658	0.9066	0.7728	0.9066	0.7655	0.8096	**0.7916**	0.8096	**0.7916**	0.8087	0.7861	0.8407	0.7836
Ecoli046vs5	0.9111	0.9250	0.9781	0.9282	0.9754	0.9282	0.9328	0.9365	0.9328	0.9365	0.9172	**0.9450**	0.9335	0.9392
Ecoli01vs235	0.7733	0.7700	0.8705	0.7936	0.8705	0.7936	0.8099	0.7850	0.8926	0.8827	0.8973	**0.8959**	0.9181	0.8564
Ecoli0267vs35	0.7263	0.7725	0.8629	0.8401	0.8604	0.8327	0.7745	0.8026	0.8516	0.8526	0.8491	**0.8677**	0.8566	0.8150
Glass04vs5	0.8702	0.9441	0.9412	0.9632	0.9397	0.9511	0.9789	**0.9941**	0.9789	**0.9941**	0.9517	0.9761	0.9789	**0.9941**
Ecoli0346vs5	0.8368	0.8000	0.8791	0.8169	0.8791	0.8169	0.8434	0.8419	0.9282	0.9419	0.9096	0.9095	0.9329	**0.9446**
Ecoli0347vs56	0.7925	0.7735	0.8623	0.7920	0.5000	0.5000	0.8263	0.8363	0.9109	0.9119	0.9089	0.9212	0.9317	**0.9227**
Yeast05679vs4	0.6288	0.6257	0.8954	0.7440	0.8978	0.7682	0.7443	**0.7968**	0.7443	**0.7968**	0.7383	0.7915	0.8557	0.7825
Ecoli067vs5	0.8031	0.8225	0.9500	0.8375	0.9456	0.8250	0.8769	0.8675	0.8769	0.8675	0.8750	**0.8825**	0.8863	0.8600
Vowel0	0.9915	0.9939	0.9999	**0.9994**	0.9999	**0.9994**	0.9975	**0.9994**	0.9975	**0.9994**	0.9808	0.9800	0.9975	**0.9994**
Glass016vs2	0.5629	0.6357	0.8800	0.7169	0.8771	0.6445	0.7477	**0.7893**	0.7477	**0.7893**	0.7640	0.7864	0.7982	0.7560
Glass2	0.5474	0.5302	0.9150	0.7162	0.8984	**0.7717**	0.6969	0.6954	0.6969	0.6954	0.7254	0.7334	0.8470	0.6733
Ecoli0147vs2356	0.7838	0.7968	0.8605	0.7959	0.8609	0.7959	0.8160	0.8272	0.8969	0.9057	0.8907	0.9041	0.9262	**0.9170**
Led7digit02456789vs1	0.7696	0.7747	0.8618	0.8215	0.8642	0.8465	0.8261	0.8297	0.8261	0.8297	0.8311	0.8223	0.9018	**0.8639**
Glass06vs5	0.8725	0.9500	0.9786	0.9847	0.9786	0.9847	0.9240	**1.0000**	0.9240	**1.0000**	0.9205	0.9400	0.9383	**1.0000**
Ecoli01vs5	0.8932	0.9000	0.9739	0.9023	0.9733	0.9023	0.9216	0.9136	0.9216	0.9136	0.9239	0.9068	0.9312	**0.9159**
Glass0146vs2	0.5302	0.5727	0.8903	0.7019	0.8923	0.7018	0.6940	**0.7567**	0.6940	**0.7567**	0.7339	0.7458	0.7404	0.6447
Ecoli0147vs56	0.8793	0.8551	0.9666	0.9139	0.9601	0.9025	0.9221	0.9189	0.9221	0.9189	0.9238	0.9156	0.9340	**0.9254**
Cleveland0vs4	0.7726	0.7136	0.9320	0.8346	0.9320	0.8346	0.8487	**0.8584**	0.8487	**0.8584**	0.8448	0.8553	0.8727	0.8583
Ecoli0146vs5	0.9058	0.9231	0.9740	0.9019	0.9745	0.9000	0.9168	0.9135	0.9168	0.9135	0.9159	**0.9250**	0.9197	0.9192
Ecoli4	0.8238	0.7734	0.8865	0.8421	0.8846	0.8108	0.8366	0.8187	0.9163	0.9155	0.9217	0.9107	0.9281	**0.9202**
Yeast1vs7	0.6153	0.6109	0.8802	0.7390	0.8811	0.6998	0.7170	**0.7453**	0.7170	**0.7453**	0.7175	0.7406	0.8039	0.6177
Shuttle0vs4	0.9959	0.9960	1.0000	0.9960	1.0000	0.9960	0.9959	0.9960	0.9959	0.9960	0.9959	**1.0000**	0.9959	0.9960
Glass4	0.7628	0.8425	0.9689	0.8917	0.9627	**0.9151**	0.8885	0.8868	0.8885	0.8868	0.8835	0.8868	0.8912	0.8843
Page-blocks13vs2	0.9724	0.9433	0.9963	0.9978	0.9963	**0.9989**	0.9963	0.9977	0.9963	0.9977	0.9859	0.9888	0.9972	0.9977
Abalone9vs18	0.5987	0.6332	0.9099	0.7525	0.9023	0.7416	0.6990	**0.7637**	0.7998	0.7334	0.8097	0.7408	0.7117	0.6482
Glass016vs5	0.9121	0.8971	0.9686	0.9271	0.9664	0.9186	0.9871	**0.9857**	0.9871	**0.9857**	0.9757	0.9686	0.9850	**0.9857**
Shuttle2vs4	0.8750	0.9500	0.9959	**1.0000**	0.9959	**1.0000**	0.9600	0.9500	0.9600	0.9500	0.9078	0.9140	0.9600	0.9500
Yeast1458vs7	0.5163	0.5144	0.8852	**0.6944**	0.8812	0.6929	0.6249	0.6609	0.6249	0.6609	0.6228	0.6654	0.6719	0.5729
Glass5	0.8439	0.8976	0.9780	0.9378	0.9689	**0.9732**	0.9717	0.9329	0.9717	0.9329	0.9799	0.9256	0.9580	0.9280
east2vs8	0.7236	0.7239	0.9656	0.7208	0.9608	0.7371	0.7930	**0.8012**	0.7930	**0.8012**	0.7846	**0.8012**	0.8131	0.7631
Yeast4	0.5966	0.5947	0.9594	0.7444	0.9520	0.7571	0.7281	0.7489	0.7281	0.7489	0.7279	0.7489	0.8787	**0.7708**
Yeast1289vs7	0.5520	0.5484	0.9185	0.6586	0.9170	**0.6764**	0.6677	0.6462	0.6677	0.6462	0.6671	0.6629	0.7135	0.6154
Yeast5	0.8056	0.8128	0.9836	0.9503	0.9812	**0.9566**	0.9357	0.9424	0.9357	0.9424	0.9394	0.9389	0.9530	0.9174
Ecoli0137vs26	0.7730	0.7982	0.8680	0.7691	0.5000	0.5000	0.7607	0.7800	0.8361	0.8281	0.8293	0.8244	0.8516	**0.8445**
Yeast6	0.7570	0.7527	0.9720	0.8442	0.9676	0.8540	0.8145	0.8368	0.8145	0.8368	0.8198	0.8497	0.8890	**0.8678**
Abalone19	0.4998	0.4998	0.9780	0.5216	0.9737	0.5205	0.5402	0.5184	0.7576	0.5193	0.7573	**0.5357**	0.6215	0.5114
Average	0.7697	0.7752	0.8880	0.8212	0.8743	0.8166	0.8229	0.8295	0.8594	**0.8596**	0.8564	0.8561	0.8849	0.8509

results for the C4.5 algorithm versions. Next, the results for the SVM versions used in the study are shown in Table A.16. Later, the results for the FH-GBML algorithm versions are presented in Table A.17. Finally, Table A.18 show the average results for each dataset for the 3-NN algorithm. We stress in **boldface** the best results achieved by a version.

References

Akbani, R., Kwek, S., & Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. In *ECML* (pp. 39–50).

Alaiz-Rodríguez, R., Guerrero-Curieses, A., & Cid-Sueiro, J. (2009). Improving classification under changes in class and within-class distributions. In *IWANN'09: Proceedings of the 10th international work-*

conference on artificial neural networks (pp. 122–130). Berlin, Heidelberg: Springer.
Alaiz-Rodríguez, R., & Japkowicz, N. (2008). Assessing the impact of changing environments on classifier performance. In *Canadian AI'08: Proceedings of the Canadian society for computational studies of intelligence, 21st conference on advances in artificial intelligence* (pp. 13–24). Berlin, Heidelberg: Springer-Verlag.
Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., et al. (2011). KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multi-Valued Logic and Soft Computing, 17*, 255–287.
Alcalá-Fdez, J., Sánchez, L., García, S., del Jesus, M. J., Ventura, S., Garrell, J. M., et al. (2008). Keel: A software tool to assess evolutionary algorithms for data mining problems. *Soft Computing, 13*, 307–318.
Barandela, R., Sánchez, J. S., García, V., & Rangel, E. (2003). Strategies for learning in class imbalance problems. *Pattern Recognition, 36*, 849–851.
Batista, G. E. A. P. A., Prati, R. C., & Monard, M. C. (2004). A study of the behaviour of several methods for balancing machine learning training data. *SIGKDD Explorations, 6*, 20–29.
Bermejo, P., Gámez, J., & Puerta, J. (2011). Improving the performance of Naive Bayes multinomial in e-mail foldering by introducing distribution-based balance of datasets. *Expert Systems with Applications, 38*, 2072–2080.
Bickel, S., Brückner, M., & Scheffer, T. (2009). Discriminative learning under covariate shift. *Journal of Machine Learning Research, 10*, 2137–2155.
Bradford, J. P., Kunz, C., Kohavi, R., Brunk, C., & Brodley, C. E. (1998). Pruning decision trees with misclassification costs. In *ECML* (pp. 131–136).
Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition, 30*, 1145–1159.
Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2009). Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Proceedings of the 13th Pacific-Asia conference on advances in knowledge discovery and data mining PAKDD'09* (pp. 475–482).
Quiñonero Candela, J., Sugiyama, M., Schwaighofer, A., & Lawrence, N. D. (2009). *Dataset shift in machine learning*. The MIT Press.
Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991). Fuzzy art: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks, 4*, 759–771.
Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligent Research, 16*, 321–357.
Chawla, N. V., Cieslak, D. A., Hall, L. O., & Joshi, A. (2008). Automatically countering imbalance and its empirical relationship to cost. *Data Mining and Knowledge Discovery, 17*, 225–252.
Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Editorial: Special issue on learning from imbalanced data sets. *SIGKDD Explorations, 6*, 1–6.
Chen, M.-C., Chen, L.-S., Hsu, C.-C., & Zeng, W.-R. (2008). An information granulation based data mining approach for classifying imbalanced data. *Information Sciences, 178*, 3214–3227.
Chen, X., Fang, T., Huo, H., & Li, D. (2011). Graph-based feature selection for object-oriented classification in vhr airborne imagery. *IEEE Transactions on Geoscience and Remote Sensing, 49*, 353–365.
Cieslak, D. A., & Chawla, N. V. (2009). A framework for monitoring classifiers performance: When and why failure occurs? *Knowledge and Information Systems, 18*, 83–108.
Cortes, C., & Vapnik, V. (1995). Support vector networks. *Machine Learning, 20*, 273–297.
Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*, 21–27.
Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research, 7*, 1–30.
Denil, M., & Trappenberg, T. (2010). Overlap versus imbalance. In *Canadian AI 2010, LNAI* (Vol. 6085, pp. 220–231).
Domingos, P. (1999). Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the 5th international conference on knowledge discovery and data mining (KDD'99)* (pp. 155–164).
Ducange, P., Lazzerini, B., & Marcelloni, F. (2010). Multi-objective genetic fuzzy classifiers for imbalanced and cost-sensitive datasets. *Soft Computing, 14*, 713–728.
Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceedings of the 17th IEEE international joint conference on artificial intelligence (IJCAI'01)* (pp. 973–978).
Estabrooks, A., Jo, T., & Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence, 20*, 18–36.
Fernández, A., García, S., del Jesus, M. J., & Herrera, F. (2008). A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. *Fuzzy Sets and Systems, 159*, 2378–2398.
Fernández, A., del Jesus, M. J., & Herrera, F. (2009). Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets. *International Journal of Approximate Reasoning, 50*, 561–577.
Fernández, A., del Jesús, M. J., & Herrera, F. (2010). On the 2-tuples based genetic tuning performance for fuzzy rule based classification systems in imbalanced data-sets. *Information Sciences, 180*, 1268–1291.
Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association, 32*, 675–701.
Gama, J. (2004). Functional trees. *Machine Learning, 55*, 219–250.
García, S., Fernández, A., Luengo, J., & Herrera, F. (2009). A study of statistical techniques and performance measures for genetics-based machine learning: Accuracy and interpretability. *Soft Computing, 13*, 959–977.
García, S., Fernández, A., Luengo, J., & Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences, 180*, 2044–2064.
García, S., & Herrera, F. (2008). An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research, 9*, 2607–2624.
García, V., Mollineda, R., & Sánchez, J. S. (2008). On the k-NN performance in a challenging scenario of imbalance and overlapping. *Pattern Analysis Applications, 11*, 269–280.
Globerson, A., Teo, C. H., Smola, A., & Roweis, S. (2009). An adversarial view of covariate shift and a minimax approach. In J. Quiñonero Candela, M. Sugiyama, A. Schwaighofer, & N. D. Lawrence (Eds.), *Dataset shift in machine learning* (pp. 179–198). The MIT Press.
Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., & Schölkopf, B. (2009). Covariate shift by kernel mean matching. In J. Quiñonero Candela, M. Sugiyama, A. Schwaighofer, & N. D. Lawrence (Eds.), *Dataset shift in machine learning* (pp. 131–160). The MIT Press.
Han, H., Wang, W., & Mao, B. (2005). Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *Proceedings of the international conference on intelligent computing* (pp. 878–887).
Hand, D. J., & Vinciotti, V. (2003). Choosing k for two-class nearest neighbour classifiers with unbalanced classes. *Pattern Recognition Letters, 24*, 1555–1562.
He, H., Bai, Y., Garcia, E., & Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *Proceedings of international joint conference on neural networks* (pp. 1322–1328).
He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering, 21*, 1263–1284.
Ho, T., & Basu, M. (2002). Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*, 289–300.
Huang, J., & Ling, C. X. (2005). Using auc and accuracy in evaluating learning algorithms – appendices. *IEEE Transactions on Knowledge and Data Engineering*, 17.
Ishibuchi, H., & Nakashima, T. (2001). Effect of rule weights in fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems, 9*, 506–515.
Ishibuchi, H., & Yamamoto, T. (2005). Rule weight specification in fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems, 13*, 428–435.
Ishibuchi, H., Yamamoto, T., & Nakashima, T. (2005). Hybridization of fuzzy GBML approaches for pattern classification problems. *IEEE Transactions on System, Man and Cybernetics B, 35*, 359–365.
Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis Journal, 6*, 429–450.
Kaburlasos, V. G. (2007). Granular enhancement of fuzzy art/som neural classifiers based on lattice theory. In *Computational intelligence based on lattice theory* (pp. 3–23).
Kubat, M., & Matwin, S. (1997). Addressing the curse of imbalanced training sets: One-sided selection. In *International conference on machine learning* (pp. 179–186).
Laurikkala, J. (2001). Improving identification of difficult small classes by balancing class distribution. In *Proceedings of the conference on AI in medicine in Europe: Artificial intelligence medicine* (pp. 63–66).
Ling, C. X., Yang, Q., Wang, J., & Zhang, S. (2004). Decision trees with minimal costs. In *ICML*.
López, V., Fernández, A., & Herrera, F. (2010). A first approach for cost-sensitive classification with linguistic genetic fuzzy systems in imbalanced data-sets. In *10th International conference on intelligent systems design and applications, ISDA 2010* (pp. 676–681).
Luengo, J., Fernández, A., García, S., & Herrera, F. (2011). Addressing data complexity for imbalanced data sets: Analysis of SMOTE-based oversampling and evolutionary undersampling. *Soft Computing, 15*, 1909–1936.
Martín-Félez, R., & Mollineda, R. (2010). On the suitability of combining feature selection and resampling to manage data complexity. In *CAEPIA 2009, LNAI* (Vol. 5988, pp. 141–150).
Moreno-Torres, J. G., & Herrera, F. (2010). A preliminary study on overlapping and data fracture in imbalanced domains by means of genetic programming-based feature extraction. In *Proceedings of the 10th international conference on intelligent systems design and applications (ISDA 2010)* (pp. 501–506).
Moreno-Torres, J. G., Llorà, X., Goldberg, D. E., & Bhargava, R. (in press). Repairing fractures between data using genetic programming-based feature extraction: A case study in cancer diagnosis. *Information Sciences*, doi:10.1016/j.ins.2010.09.018.
Orriols-Puig, A., & Bernadó-Mansilla, E. (2009). Evolutionary rule-based systems for imbalanced datasets. *Soft Computing, 13*, 213–225.
Prati, R. C., & Batista, G. E. A. P. A. (2004). Class imbalances versus class overlapping: An analysis of a learning system behavior. In *Proceedings of Mexican international conference on artificial intelligence (MICAI)* (pp. 312–321).
Quinlan, J. (1993). *C4.5: Programs for machine learning*. Morgan Kauffman.
Riddle, P., Segal, R., & Etzioni, O. (1994). Representation design and brute-force induction in a boeing manufacturing domain. *Applied Artificial Intelligence, 8*, 125–147.

Shaffer, J. (1986). Modified sequentially rejective multiple test procedures. *Journal of the American Statistical Association, 81*, 826–831.
Sheskin, D. (2006). *Handbook of parametric and nonparametric statistical procedures*. Chapman & Hall.
Shimodaira, H. (2000). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference, 90*, 227–244.
Sikora, M. (2011). Induction and pruning of classification rules for prediction of microseismic hazards in coal mines. *Expert Systems with Applications, 38*, 6748–6758.
Sugiyama, M., Krauledat, M., & Müller, K.-R. (2007). Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research, 8*, 985–1005.
Sun, Y., Kamel, M. S., Wong, A. K. C., & Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition, 40*, 3358–3378.
Sun, Y., Wong, A. K. C., & Kamel, M. S. (2009). Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence, 23*, 687–719.
Ting, K. M. (2002). An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering, 14*, 659–665.
Vapnik, V. (1998). *Statistical learning theory*. New York, USA: Wiley.
Vapnik, V., & Lerner, A. (1963). Pattern recognition using generalized portrait method. *Automation and Remote Control, 24*.
Veropoulos, K., Campbell, C., & Cristianini, N. (1999). Controlling the sensitivity of support vector machines. In *Proceedings of the international joint conference on AI* (pp. 55–60).
Villar, P., Fernández, A., & Herrera, F. (2010). A genetic algorithm for feature selection and granularity learning in fuzzy rule-based classification systems for highly imbalanced data-sets. In *13th International conference on information processing and management of uncertainty in knowledge-based systems (IPMU2010), CCIS* (Vol. 80, pp. 741–750).
Villar, P., Fernández, A., Sánchez, A., & Herrera, F. (2009). Un algoritmo genético para selección de características en sistemas de clasificación basados en reglas difusas para conjuntos de datos altamente no balanceados. In *Actas de la XIII Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA09)* (pp. 519–528).
Wang, B., & Japkowicz, N. (2004). Imbalanced data set learning with synthetic samples. In *Proceedings of IRIS machine learning workshop*.
Wang, K., Zhou, S., Fu, C. A., Yu, J. X., Jeffrey, F., & Yu, X. (2003). Mining changes of classification by correspondence tracing. In *Proceedings of the 2003 SIAM international conference on data mining (SDM 2003)*.
Weiss, G. M., & Provost, F. J. (2003). Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research, 19*, 315–354.
Wilson, D. L. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man and Cybernetics, 2*, 408–421.
Wu, G., & Chang, E. Y. (2005). Kba: Kernel boundary alignment considering imbalanced data distribution. *IEEE Transactions on Knowledge and Data Engineering, 17*, 786–795.
Xu, L., Chow, M.-Y., & Taylor, L. S. (2007). Power distribution fault cause identification with imbalanced data using the data mining-based fuzzy classification e-algorithm. *IEEE Transactions on Power Systems, 22*, 164–171.
Yang, Q., & Wu, X. (2006). 10 Challenging problems in data mining research. *International Journal of Information Technology and Decision Making, 5*, 597–604.
Yang, Y., Wu, X., & Zhu, X. (2008). Conceptual equivalence for contrast mining in classification learning. *Data and Knowledge Engineering, 67*, 413–429.
Zadrozny, B., & Elkan, C. (2001). Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the 7th international conference on knowledge discovery and data mining (KDD'01)* (pp. 204–213).
Zadrozny, B., Langford, J., & Abe, N. (2003). Cost-sensitive learning by cost-proportionate example weighting. In *Proceedings of the 3rd IEEE international conference on data mining (ICDM'03)* (pp. 435–442).
Zhou, Z.-H., & Liu, X.-Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering, 18*, 63–77.

1.2. An Insight into Classification with Imbalanced Data: Empirical Results and Current Trends on Using Data Intrinsic Characteristics

- V. López, A. Fernández, S. García, V. Palade, F. Herrera, An Insight into Classification with Imbalanced Data: Empirical Results and Current Trends on Using Data Intrinsic Characteristics. Information Sciences 250 (2013) 113–141, doi: 10.1016/j.ins.2013.07.007

  - Status: **Published**.
  - Impact Factor (JCR 2012): 3.643.
  - Subject Category: Computer Science, Information Systems. Ranking 6 / 132 (**Q1**).

Information Sciences 250 (2013) 113–141

Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics

Victoria López [a,*], Alberto Fernández [b], Salvador García [b], Vasile Palade [c], Francisco Herrera [a]

[a] Dept. of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, Granada, Spain
[b] Dept. of Computer Science, University of Jaén, Jaén, Spain
[c] Department of Computer Science, University of Oxford, Oxford OX1 3QD, United Kingdom

ARTICLE INFO

Article history:
Received 2 October 2012
Received in revised form 16 April 2013
Accepted 5 July 2013
Available online 12 July 2013

Keywords:
Imbalanced dataset
Sampling
Cost-sensitive learning
Small disjuncts
Noisy data
Dataset shift

ABSTRACT

Training classifiers with datasets which suffer of imbalanced class distributions is an important problem in data mining. This issue occurs when the number of examples representing the class of interest is much lower than the ones of the other classes. Its presence in many real-world applications has brought along a growth of attention from researchers.

We shortly review the many issues in machine learning and applications of this problem, by introducing the characteristics of the imbalanced dataset scenario in classification, presenting the specific metrics for evaluating performance in class imbalanced learning and enumerating the proposed solutions. In particular, we will describe preprocessing, cost-sensitive learning and ensemble techniques, carrying out an experimental study to contrast these approaches in an intra and inter-family comparison.

We will carry out a thorough discussion on the main issues related to using data intrinsic characteristics in this classification problem. This will help to improve the current models with respect to: the presence of small disjuncts, the lack of density in the training data, the overlapping between classes, the identification of noisy data, the significance of the borderline instances, and the dataset shift between the training and the test distributions. Finally, we introduce several approaches and recommendations to address these problems in conjunction with imbalanced data, and we will show some experimental examples on the behavior of the learning algorithms on data with such intrinsic characteristics.

1. Introduction

In many supervised learning applications, there is a significant difference between the prior probabilities of different classes, i.e., between the probabilities with which an example belongs to the different classes of the classification problem. This situation is known as the class imbalance problem [29,66,118] and it is common in many real problems from telecommunications, web, finance-world, ecology, biology, medicine not only, and which can be considered one of the top problems in data mining today [143]. Furthermore, it is worth to point out that the minority class is usually the one that has the highest interest from a learning point of view and it also implies a great cost when it is not well classified [42].

The hitch with imbalanced datasets is that standard classification learning algorithms are often biased towards the majority class (known as the "negative" class) and therefore there is a higher misclassification rate for the minority class instances

* Corresponding author. Tel.: +34 953 213016; fax: +34 953 212472.
E-mail addresses: vlopez@decsai.ugr.es (V. López), alberto.fernandez@ujaen.es (A. Fernández), sglopez@ujaen.es (S. García), vasile.palade@cs.ox.ac.uk (V. Palade), herrera@decsai.ugr.es (F. Herrera).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.07.007

(called the "positive" examples). Therefore, throughout the last years, many solutions have been proposed to deal with this problem, both for standard learning algorithms and for ensemble techniques [50]. They can be categorized into three major groups:

1. **Data sampling**: In which the training instances are modified in such a way to produce a more or less balanced class distribution that allow classifiers to perform in a similar manner to standard classification [9,27].
2. **Algorithmic modification**: This procedure is oriented towards the adaptation of base learning methods to be more attuned to class imbalance issues [147]
3. **Cost-sensitive learning**: This type of solutions incorporate approaches at the data level, at the algorithmic level, or at both levels combined, considering higher costs for the misclassification of examples of the positive class with respect to the negative class, and therefore, trying to minimize higher cost errors [38,148].

In this paper, our first goal is to come up with a review on this type of methodologies, presenting a taxonomy for each group, enumerating and briefly describing the main properties of the most significant approaches that have been traditionally applied in this field. Furthermore, we carry out an experimental study in order to highlight the behavior of the different paradigms that were previously presented.

Most of the studies on the behavior of several standard classifiers in imbalance domains have shown that significant loss of performance is mainly due to the skewed class distribution, given by the imbalance ratio (IR), defined as the ratio of the number of instances in the majority class to the number of examples in the minority class [58,98]. However, there are several investigations which also suggest that there are other factors that contribute to such performance degradation [72]. Therefore, as a second goal, we present a discussion about six significant problems related to data intrinsic characteristics and that must be taken into account in order to provide better solutions for correctly identifying both classes of the problem:

1. The identification of areas with small disjuncts [136,137].
2. The lack of density and information in the training data [133].
3. The problem of overlapping between the classes [37,55].
4. The impact of noisy data in imbalanced domains [20,111].
5. The significance of the borderline instances to carry out a good discrimination between the positive and negative classes, and its relationship with noisy examples [39,97].
6. The possible differences in the data distribution for the training and test data, also known as the dataset shift [95,114].

This thorough study of the problem can guide us about the source where the difficulties for imbalanced classification emerge, focusing on the analysis of significant data intrinsic characteristics. Specifically, for each established scenario we show an experimental example on how it affects the behavior of the learning algorithms, in order to stress its significance.

We must point out that some of these topics have recent studies associated, which are described along this paper, examining their main contributions and recommendations. However, we emphasize that they still need to be addressed in more detail in order to have models of high quality in this classification scenario and, therefore, we have stressed them as future trends of research for imbalanced learning. Overcoming these problems can be the key for developing new approaches that improve the correct identification of both the minority and majority classes.

In summary, the main contributions of this new review on former works on this topic [66,118] can be highlighted with respect to two points: (1) the extensive experimental study with a large benchmark of 66 imbalanced datasets for analyzing the behavior of the solutions proposed to address the problem of imbalanced data; and (2) a detailed analysis and study of the data intrinsic characteristics in this scenario and a brief description on how they affect the performance of the classification algorithms.

With this aim in mind, this paper is organized as follows. First, Section 2 presents the problem of imbalanced datasets, introducing its features and the metrics employed in this context. Section 3 describes the diverse preprocessing, cost-sensitive learning and ensemble methodologies that have been proposed to deal with this problem. Next, we develop an experimental study for contrasting the behavior of these approaches in Section 4. Section 5 is devoted to analyzing and discussing the aforementioned problems associated with data intrinsic characteristics. Finally, Section 6 summarizes and concludes the work.

2. Imbalanced datasets in classification

In this section, we first introduce the problem of imbalanced datasets and then we present the evaluation metrics for this type of classification problem, which differ from usual measures in classification.

2.1. The problem of imbalanced datasets

In the classification problem field, the scenario of imbalanced datasets appears frequently. The main property of this type of classification problem is that the examples of one class significantly outnumber the examples of the other one [66,118].

The minority class usually represents the most important concept to be learned, and it is difficult to identify it since it might be associated with exceptional and significant cases [135], or because the data acquisition of these examples is costly [139]. In most cases, the imbalanced class problem is associated to binary classification, but the multi-class problem often occurs and, since there can be several minority classes, it is more difficult to solve [48,81].

Since most of the standard learning algorithms consider a balanced training set, this may generate suboptimal classification models, i.e. a good coverage of the majority examples, whereas the minority ones are misclassified frequently. Therefore, those algorithms, which obtain a good behavior in the framework of standard classification, do not necessarily achieve the best performance for imbalanced datasets [47]. There are several reasons behind this behavior:

1. The use of global performance measures for guiding the learning process, such as the standard accuracy rate, may provide an advantage to the majority class.
2. Classification rules that predict the positive class are often highly specialized and thus their coverage is very low, hence they are discarded in favor of more general rules, i.e. those that predict the negative class.
3. Very small clusters of minority class examples can be identified as noise, and therefore they could be wrongly discarded by the classifier. On the contrary, few real noisy examples can degrade the identification of the minority class, since it has fewer examples to train with.

In recent years, the imbalanced learning problem has received much attention from the machine learning community. Regarding real world domains, the importance of the imbalance learning problem is growing, since it is a recurring issue in many applications. As some examples, we could mention very high resolution airbourne imagery [31], forecasting of ozone levels [125], face recognition [78], and especially medical diagnosis [11,86,91,93,132]. It is important to remember that the minority class usually represents the concept of interest and it is the most difficult to obtain from real data, for example patients with illnesses in a medical diagnosis problem; whereas the other class represents the counterpart of that concept (healthy patients).

2.2. Evaluation in imbalanced domains

The evaluation criteria is a key factor in assessing the classification performance and guiding the classifier modeling. In a two-class problem, the confusion matrix (shown in Table 1) records the results of correctly and incorrectly recognized examples of each class.

Traditionally, the accuracy rate (Eq. (1)) has been the most commonly used empirical measure. However, in the framework of imbalanced datasets, accuracy is no longer a proper measure, since it does not distinguish between the number of correctly classified examples of different classes. Hence, it may lead to erroneous conclusions, i.e., a classifier achieving an accuracy of 90% in a dataset with an IR value of 9 is not accurate if it classifies all examples as negatives.

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \tag{1}$$

In imbalanced domains, the evaluation of the classifiers' performance must be carried out using specific metrics in order to take into account the class distribution. Concretely, we can obtain four metrics from Table 1 to measure the classification performance of both, positive and negative, classes independently:

- **True positive rate**: $TP_{rate} = \frac{TP}{TP+FN}$ is the percentage of positive instances correctly classified.
- **True negative rate**: $TN_{rate} = \frac{TN}{FP+TN}$ is the percentage of negative instances correctly classified.
- **False positive rate**: $FP_{rate} = \frac{FP}{FP+TN}$ is the percentage of negative instances misclassified.
- **False negative rate**: $FN_{rate} = \frac{FN}{TP+FN}$ is the percentage of positive instances misclassified.

Since in this classification scenario we intend to achieve good quality results for both classes, there is a necessity of combining the individual measures of both the positive and negative classes, as none of these measures alone is adequate by itself.

A well-known approach to unify these measures and to produce an evaluation criteria is to use the Receiver Operating Characteristic (ROC) graphic [19]. This graphic allows the visualization of the trade-off between the benefits (TP_{rate}) and costs (FP_{rate}), as it evidences that any classifier cannot increase the number of true positives without also increasing the false positives. The Area Under the ROC Curve (*AUC*) [70] corresponds to the probability of correctly identifying which one of the two

Table 1
Confusion matrix for a two-class problem.

	Positive prediction	Negative prediction
Positive class	True Positive (TP)	False Negative (FN)
Negative class	False Positive (FP)	True Negative (TN)

stimuli is noise and which one is signal plus noise. The *AUC* provides a single measure of a classifier's performance for evaluating which model is better on average. Fig. 1 shows how to build the ROC space plotting on a two-dimensional chart the TP_{rate} (Y-axis) against the FP_{rate} (X-axis). Points in $(0,0)$ and $(1,1)$ are trivial classifiers where the predicted class is always the negative and positive one, respectively. On the contrary, $(0,1)$ point represents the perfect classifier. The*AUC* measure is computed just by obtaining the area of the graphic:

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \tag{2}$$

In [103], the significance of these graphical methods for the classification predictive performance evaluation is stressed. According to the authors, the main advantage of this type of methods resides in their ability to depict the trade-offs between evaluation aspects in a multidimensional space rather than reducing these aspects to an arbitrarily chosen (and often biased) single scalar measure. In particular, they present a review of several representation mechanisms emphasizing the best scenario for their use; for example, in imbalanced domains, when we are interested in the positive class, it is recommended the use of precision-recall graphs [36]. Furthermore, the expected cost or profit of each model might be analyzed using cost curves [40], lift and ROI graphs [83].

Other metric of interest to be stressed in this area is the geometric mean of the true rates [7], which can be defined as:

$$GM = \sqrt{\frac{TP}{TP + FN} \cdot \frac{TN}{FP + TN}} \tag{3}$$

This metric attempts to maximize the accuracy on each of the two classes with a good balance, being a performance metric that correlates both objectives. However, due to this symmetric nature of the distribution of the geometric mean over TP_{rate} (sensitivity) and the TN_{rate} (specificity), it is hard to contrast different models according to their precision on each class.

Another significant performance metric that is commonly used is the F-measure [6]:

$$\begin{aligned} F_m &= \frac{(1 + \beta^2)(PPV \cdot TP_{rate})}{\beta^2 PPV + TP_{rate}} \\ PPV &= \frac{TP}{TP + FP} \end{aligned} \tag{4}$$

A popular choice for β is 1, where equal importance is assigned for both TP_{rate} and the positive predictive value (*PPV*). This measure would be more sensitive to the changes in the *PPV* than to the changes in TP_{rate}, which can lead to the selection of sub-optimal models.

According to the previous comments, some authors try to propose several measures for imbalanced domains in order to be able to obtain as much information as possible about the contribution of each class to the final performance and to take into account the IR of the dataset as an indication of its difficulty. For example, in [10,14] the *Adjusted G-mean* is proposed. This measure is designed towards obtaining the highest sensitivity (TP_{rate}) without decreasing too much the specificity (TN_{rate}). This fact is measured with respect to the original model, i.e. the original classifier without addressing the class imbalance problem. Eq. 5 shows its definition:

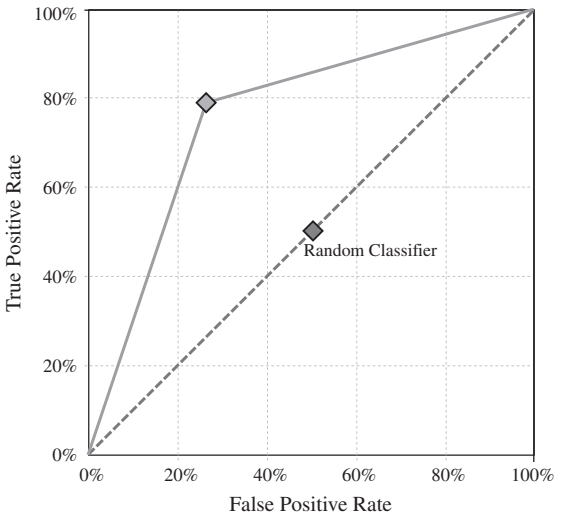


Fig. 1. Example of a ROC plot. Two classifiers' curves are depicted: the dashed line represents a random classifier, whereas the solid line is a classifier which is better than the random classifier.

$$\begin{aligned} AGM &= \frac{GM+TN_{rate}\cdot(FP+TN)}{1+FP+TN}; \quad \text{If} TP_{rate} > 0 \\ AGM &= 0; \qquad\qquad\qquad\quad \text{If} TP_{rate} = 0 \end{aligned} \tag{5}$$

Additionally, in [54] the authors presented a simple performance metric, called *Dominance*, which is aimed to point out the dominance or prevalence relationship between the positive class and the negative class, in the range $[-1,+1]$ (Eq. 6). Furthermore, it can be used as a visual tool to analyze the behavior of a classifier on a 2-D space from the joint perspective of global precision (Y-axis) and dominance (X-axis).

$$Dom = TP_{rate} - TN_{rate} \tag{6}$$

The same authors, using the previous concept of *dominance*, *Index of Balanced Accuracy* (IBA) [56,57]. IBA weights a performance measure, that aims to make it more sensitive for imbalanced domains. The weighting factor favors those results with moderately better classification rates on the minority class. IBA is formulated as follows:

$$IBA_{\alpha}(M) = (1+\alpha\cdot Dom)M \tag{7}$$

where $(1+\alpha\cdot Dom)$ is the weighting factor and M represents a performance metric. The objective is to moderately favor the classification models with higher prediction rate on the minority class (without underestimating the relevance of the majority class) by means of a weighted function of any plain performance evaluation measure.

A comparison regarding these evaluation proposals for imbalanced datasets is out of the scope of this paper. For this reason, we refer any interested reader to find a deep experimental study in [57,105].

3. Addressing classification with imbalanced data: preprocessing, cost-sensitive learning and ensemble techniques

A large number of approaches have been proposed to deal with the class imbalance problem. These approaches can be categorized into two groups: the internal approaches that create new algorithms or modify existing ones to take the class-imbalance problem into consideration [7,41,82,129,152] and external approaches that preprocess the data in order to diminish the effect of their class imbalance [9,43]. Furthermore, cost-sensitive learning solutions incorporating both the data (external) and algorithmic level (internal) approaches assume higher misclassification costs for samples in the minority class and seek to minimize the high cost errors [15,38,59,117,150]. Ensemble methods [101,108] are also frequently adapted to imbalanced domains, either by modifying the ensemble learning algorithm at the data-level approach to preprocess the data before the learning stage of each classifier [17,30,112] or by embedding a cost-sensitive framework in the ensemble learning process [44,117,122].

Regarding this, in this section we first introduce the main aspects of the preprocessing techniques. Next, we describe the cost-sensitive learning approach. Finally, we present some relevant ensemble techniques in the framework of imbalanced datasets.

3.1. Preprocessing imbalanced datasets: resampling techniques

In the specialized literature, we can find some papers about resampling techniques studying the effect of changing the class distribution in order to deal with imbalanced datasets.

Those works have proved empirically that applying a preprocessing step in order to balance the class distribution is usually an useful solution [9,12,45,46]. Furthermore, the main advantage of these techniques is that they are independent of the underlying classifier.

Resampling techniques can be categorized into three groups or families:

1. *Undersampling methods*, which create a subset of the original dataset by eliminating instances (usually majority class instances).
2. *Oversampling methods*, which create a superset of the original dataset by replicating some instances or creating new instances from existing ones.
3. *Hybrids methods*, which combine both sampling approaches from above.

Within these families of methods, the simplest preprocessing techniques are non-heuristic methods such as random undersampling and random oversampling. In the first case, the major drawback is that it can discard potentially useful data, that could be important for the learning process. For random oversampling, several authors agree that this method can increase the likelihood of occurring overfitting, since it makes exact copies of existing instances.

In order to deal with the mentioned problems, more sophisticated methods have been proposed. Among them, the "Synthetic Minority Oversampling TEchnique" (SMOTE) [27] has become one of the most renowned approaches in this area. In brief, its main idea is to create new minority class examples by interpolating several minority class instances that lie together for oversampling the training set.

With this technique, the positive class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen. This process is illustrated in

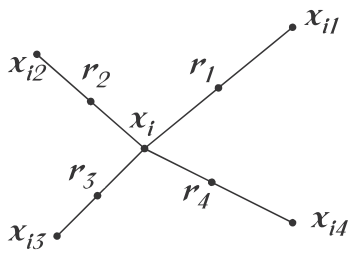


Fig. 2. An illustration of how to create the synthetic data points in the SMOTE algorithm.

Fig. 2, where x_i is the selected point, x_{i1} to x_{i4} are some selected nearest neighbors and r_1 to r_4 the synthetic data points created by the randomized interpolation.

However, in oversampling techniques, and especially for the SMOTE algorithm, the problem of over generalization is largely attributed to the way in which synthetic samples are created. Precisely, SMOTE generates the same number of synthetic data samples for each original minority example and does so without consideration to neighboring examples, which increases the occurrence of overlapping between classes [128]. To this end, various adaptive sampling methods have been proposed to overcome this limitation; some representative works include the Borderline-SMOTE [63], Adaptive Synthetic Sampling [65], Safe-Level-SMOTE [21] and SPIDER2 [116] algorithms.

Regarding undersampling, most of the proposed approaches are based on data cleaning techniques. Some representative works in this area include the Wilson's edited nearest neighbor (ENN) [140] rule, which removes examples that differ from two of its three nearest neighbors, the one-sided selection (OSS) [76], an integration method between the condensed nearest neighbor rule [64] and Tomek Links [124] and the neighborhood cleaning rule [79], which is based on the ENN technique. Additionally, the NearMiss-2 method [149] selects the majority class examples whose average distance to the three farthest minority class examples is the smallest, and in [5] the authors proposed a method that removes the majority instances far from the decision boundaries. Furthermore, a Support Vector Machine (SVM) [35] may be used to discard redundant or irrelevant majority class examples [119]. Finally, the combination of preprocessing of instances with data cleaning techniques could lead to diminishing the overlapping that is introduced by sampling methods, i.e. the integrations of SMOTE with ENN and SMOTE with Tomek links [9]. This behavior is also present in a wrapper technique introduced in [28] that defines the best percentage to perform both undersampling and oversampling.

On the other hand, these techniques are not only carried out by means of a "neighborhood', but we must also stress some cluster-based sampling algorithms, all of which aim to organize the training data into groups with significant characteristics and then performing both undersampling and/or oversampling. Some significant examples are the Cluster-Based Oversampling (CBO) [73], Class Purity Maximization [146], Sampling-Based Clustering [145], the agglomerative Hierarchical Clustering [34] or the DBSMOTE algorithm based on DBSCAN clustering [22].

Finally, the application of genetic algorithms or particle swarm optimization for the correct identification of the most useful instances has shown to achieve good results [53,142]. Also, a training set selection can be carried out in the area of imbalanced datasets [51,52]. These methods select the best set of examples to improve the behavior of several algorithms considering for this purpose the classification performance using an appropriate imbalanced measure.

3.2. Cost-sensitive learning

Cost-sensitive learning takes into account the variable cost of a misclassification with respect to the different classes [38,148]. In this case, a cost matrix codifies the penalties $C(i,j)$ of classifying examples of one class i as a different one j, as illustrated in Table 2.

These misclassification cost values can be given by domain experts, or can be learned via other approaches [117,118]. Specifically, when dealing with imbalanced problems, it is usually more interesting to recognize the positive instances rather than the negative ones. Therefore, the cost when misclassifying a positive instance must be higher than the cost of misclassifying a negative one, i.e. $C(+,-) > C(-,+)$.

Given the cost matrix, an example should be classified into the class that has the lowest expected cost, which is known as the minimum expected cost principle. The expected cost $R(i|x)$ of classifying an instance x into class i (by a classifier) can be expressed as:

Table 2
Example of a cost matrix for a fraud detection classification problem.

	Fraudulent	Legitimate
Refuse	20$	−20$
Approve	−100$	50$

$$R(i|x) = \sum_j P(j|x) \cdot C(i,j) \tag{8}$$

where $P(j|x)$ is the probability estimation of classifying an instance into class j. That is, the classifier will classify an instance x into positive class if and only if:

$$P(0|x) \cdot C(1,0) + P(1|x) \cdot C(1,1) \leqslant P(0|x) \cdot C(0,0) + P(1|x) \cdot C(0,1)$$

or, which is equivalent:

$$P(0|x) \cdot (C(1,0) - C(0,0)) \leqslant P(1|x)(C(0,1) - C(1,1))$$

Therefore, any given cost-matrix can be converted to one with $C(0,0) = C(1,1) = 0$. Under this assumption, the classifier will classify an instance x into positive class if and only if:

$$P(0|x) \cdot C(1,0) \leqslant P(1|x) \cdot C(0,1)$$

As $P(0|x) = 1 - P(1|x)$, we can obtain a threshold p^* for the classifier to classify an instance x into positive if $P(1|x) \geqslant p^*$, where

$$p^* = \frac{C(1,0)}{C(1,0) - C(0,1)} = \frac{FP}{FP + FN} \tag{9}$$

Another possibility is to "rebalance" the original training examples the ratio of:

$$p(1)FN : p(0)FP \tag{10}$$

where $p(1)$ and $p(0)$ are the prior probability of the positive and negative examples in the original training set.

In summary, two main general approaches have been proposed to deal with cost-sensitive problems:

1. **Direct methods**: The main idea of building a direct cost-sensitive learning algorithm is to directly introduce and utilize misclassification costs into the learning algorithms.
   For example, in the context of decision tree induction, the tree-building strategies are adapted to minimize the misclassification costs. The cost information is used to: (1) choose the best attribute to split the data [84,107]; and (2) determine whether a subtree should be pruned [18]. On the other hand, other approaches based on genetic algorithms can incorporate misclassification costs in the fitness function [126].
2. **Meta-learning**: This methodology implies the integration of a "preprocessing" mechanism for the training data or a "postprocessing" of the output, in such a way that the original learning algorithm is not modified. Cost-sensitive meta-learning can be further classified into two main categories: *thresholding* and *sampling*, which are based on expressions (9) and (10) respectively:
   - **Thresholding** is based on the basic decision theory that assigns instances to the class with minimum expected cost. For example, a typical decision tree for a binary classification problem assigns the class label of a leaf node depending on the majority class of the training samples that reach the node. A cost-sensitive algorithm assigns the class label to the node that minimizes the classification cost [38,147].
   - **Sampling** is based on modifying the training dataset. The most popular technique lies in resampling the original class distribution of the training dataset according to the cost decision matrix by means of undersampling/oversampling [148] or assigning instance weights [123]. These modifications have shown to be effective and can also be applied to any cost insensitive learning algorithm [150].

3.3. Ensemble methods

Ensemble-based classifiers, also known as multiple classifier systems [101], try to improve the performance of single classifiers by inducing several classifiers and combining them to obtain a new classifier that outperforms every one of them. Hence, the basic idea is to construct several classifiers from the original data and then aggregate their predictions when unknown instances are presented.

In recent years, ensembles of classifiers have arisen as a possible solution to the class imbalance problem [77,85,112,117,127,131]. Ensemble-based methods are based on a combination between ensemble learning algorithms and one of the previously discussed techniques, namely data and algorithmic approaches, or cost-sensitive learning solutions. In the case of adding a data level approach to the ensemble learning algorithm, the new hybrid method usually preprocess the data before training each classifier. On the other hand, cost-sensitive ensembles, instead of modifying the base classifier in order to accept costs in the learning process, guide the cost minimization procedure via the ensemble learning algorithm. In this way, the modification of the base learner is avoided, but the major drawback, which is the costs definition, is still present.

A complete taxonomy for ensemble methods for learning with imbalanced classes can be found on a recent review [50], which we summarize in Fig. 3. Mainly, the authors distinguish four different families among ensemble approaches for imbalanced learning. On the one hand, they identified cost-sensitive boosting approaches which are similar to cost-sensitive

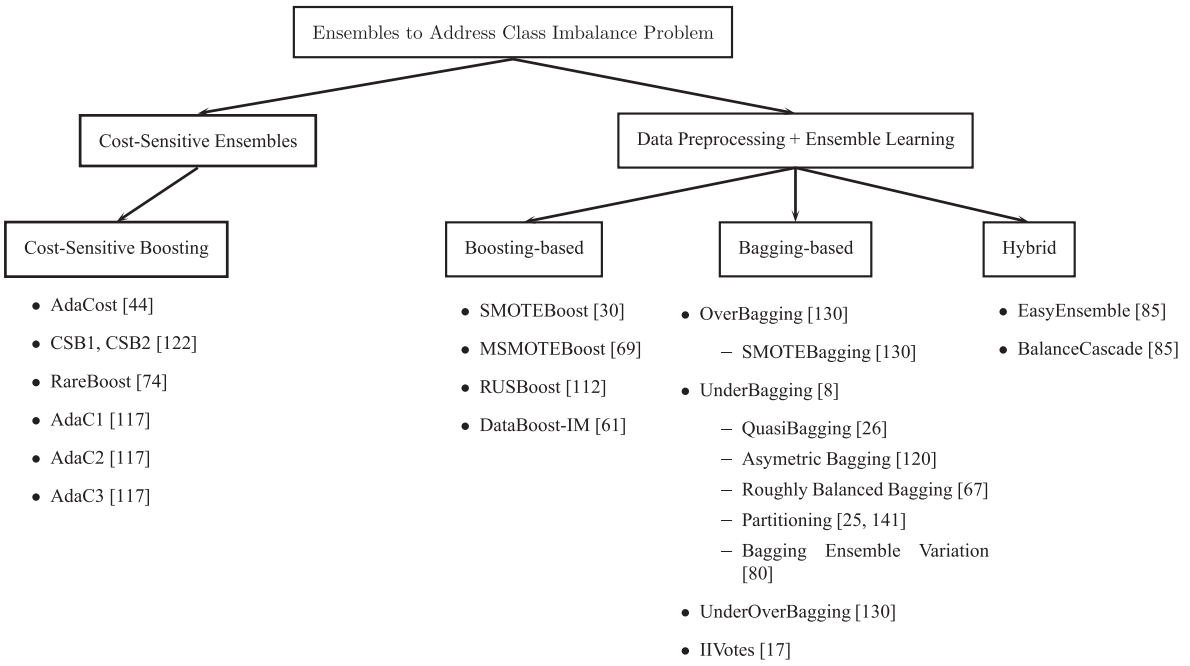


Fig. 3. Galar et al.'s proposed taxonomy for ensembles to address class imbalance problem. (See above-mentioned references for further information.)

methods, but where the costs minimization procedure is guided by a boosting algorithm. On the other hand, they distinguish three more families which have a common feature: all of them consist on embedding a data preprocessing technique in an ensemble learning algorithm. They categorized these three families depending on the ensemble learning algorithm used, i.e. boosting, bagging and hybrid ensembles.

From the study in [50], the authors concluded that ensemble-based algorithms are worthwhile, improving the results obtained by using data preprocessing techniques and training a single classifier. They also highlighted the good performance of simple approaches such as RUSBoost [112] or UnderBagging [8], which despite of being simple approaches, achieve a higher performance than many other more complex algorithms.

4. Analyzing the behavior of imbalanced learning methods

Several authors, and especially [9], have developed an ordering of the approaches to address learning with imbalanced datasets regarding a classification metric such as the AUC. In this section we present a complete study on the suitability of some recent proposals for preprocessing, cost-sensitive learning and ensemble-based methods, carrying out an intra-family comparison for selecting the best performing approaches and then developing and inter-family analysis, with the aim of observing whether there are differences among them.

In order to achieve well founded conclusions, we will make use of three classifiers based on different paradigms, namely decision trees with C4.5 [104], Support Vector Machines (SVMs) [35,100], and the well-known k-Nearest Neighbor (kNN) [92] as an Instance-Based Learning approach. The analysis will be structured in the same manner within each section: first, the average results in training and testing, together with their standard deviations, will be shown for every classifier. Then, the average rankings will be depicted in order to organize the algorithms according to their performance on the different datasets. Finally, the two highest ranked approaches will be selected for the final comparison among all the techniques.

We must remark that this study tries to be carried out in a more descriptive way. For this reason, we just carry out an "ad hoc" selection of the best approaches, even if no significant differences are found in a statistical analysis, which will be performed by means of a Shaffer post hoc test [113] ($n \times n$ comparison). Therefore, the reader must acknowledge that some of the decisions taken along this empirical analysis are carried out for the sake of simplifying our study, thus presenting an overview on the behavior of the state of the art methods on classification with imbalanced data.

According to the previous aim, we divide this section into five parts: first, in Section 4.1 we introduce the experimental framework, that is, the classification algorithms used, their parameters and the selected datasets for the study. Next, we develop a separate study for preprocessing (Section 4.2), cost-sensitive learning (Section 4.3) and and ensembles (Section 4.4). As explained earlier, the two best models will be selected as representative approaches and, finally, Section 4.5 presents a global study for the different paradigms that are analyzed.

Table 3
Summary of imbalanced datasets used.

Name	#Ex.	#Atts.	IR	Name	#Ex.	#Atts.	IR
Glass1	214	9	1.82	Glass04vs5	92	9	9.22
Ecoli0vs1	220	7	1.86	Ecoli0346vs5	205	7	9.25
Wisconsin	683	9	1.86	Ecoli0347vs56	257	7	9.28
Pima	768	8	1.90	Yeast05679vs4	528	8	9.35
Iris0	150	4	2.00	Ecoli067vs5	220	6	10.00
Glass0	214	9	2.06	Vowel0	988	13	10.10
Yeast1	1484	8	2.46	Glass016vs2	192	9	10.29
Vehicle1	846	18	2.52	Glass2	214	9	10.39
Vehicle2	846	18	2.52	Ecoli0147vs2356	336	7	10.59
Vehicle3	846	18	2.52	Led7digit02456789vs1	443	7	10.97
Haberman	306	3	2.68	Glass06vs5	108	9	11.00
Glass0123vs456	214	9	3.19	Ecoli01vs5	240	6	11.00
Vehicle0	846	18	3.23	Glass0146vs2	205	9	11.06
Ecoli1	336	7	3.36	Ecoli0147vs56	332	6	12.28
New-thyroid2	215	5	4.92	Cleveland0vs4	177	13	12.62
New-thyroid1	215	5	5.14	Ecoli0146vs5	280	6	13.00
Ecoli2	336	7	5.46	Ecoli4	336	7	13.84
Segment0	2308	19	6.01	Yeast1vs7	459	8	13.87
Glass6	214	9	6.38	Shuttle0vs4	1829	9	13.87
Yeast3	1484	8	8.11	Glass4	214	9	15.47
Ecoli3	336	7	8.19	Page-blocks13vs2	472	10	15.85
Page-blocks0	5472	10	8.77	Abalone9vs18	731	8	16.68
Ecoli034vs5	200	7	9.00	Glass016vs5	184	9	19.44
Yeast2vs4	514	8	9.08	Shuttle2vs4	129	9	20.50
Ecoli067vs35	222	7	9.09	Yeast1458vs7	693	8	22.10
Ecoli0234vs5	202	7	9.10	Glass5	214	9	22.81
Glass015vs2	172	9	9.12	Yeast2vs8	482	8	23.10
Yeast0359vs78	506	8	9.12	Yeast4	1484	8	28.41
Yeast02579vs368	1004	8	9.14	Yeast1289vs7	947	8	30.56
Yeast0256vs3789	1004	8	9.14	Yeast5	1484	8	32.78
Ecoli046vs5	203	6	9.15	Ecoli0137vs26	281	7	39.15
Ecoli01vs235	244	7	9.17	Yeast6	1484	8	39.15
Ecoli0267vs35	224	7	9.18	Abalone19	4174	8	128.87

4.1. Experimental framework

In the first place, we need to define a set of baseline classifiers to be used in all the experiments. Next, we enumerate these algorithms and also their parameter values, which have been set considering the recommendation of the corresponding authors. We must point out that these algorithms are available within the KEEL software tool [4].

1. **C4.5 Decision tree [104]**: For C4.5, we have set a confidence level of 0.25, the minimum number of item-sets per leaf was set to 2 and pruning was used as well to obtain the final tree.
2. **Support vector machines [35]**: For the SVM, we have chosen *Polykernel reference functions*, with an internal parameter of 1.0 for the exponent of each kernel function and a penalty parameter of the error term of 1.0.
3. **Instance based learning (kNN) [92]**: In this case, we have selected 1 neighbor for determining the output class, using the euclidean distance metric.

We have gathered 66 datasets, whose features are summarized in Table 3, namely the number of examples (#Ex.), number of attributes (#Atts.) and IR. Estimates of the AUC metric were obtained by means of a 5-fold cross-validation. That is, we split the dataset into 5 folds, each one containing 20% of the patterns of the dataset. For each fold, the algorithm was trained with the examples contained in the remaining folds and then tested with the current fold. This value is set up with the aim of having enough positive class instances in the different folds, hence avoiding additional problems in the data distribution [94,96], especially for highly imbalanced datasets.

We must point out that the dataset partitions employed in this paper are available for download at the KEEL dataset repository[1] [3], so that any interested researcher can use the same data for comparison.

Finally, with respect to the **evaluation metric**, we use the Area Under the ROC Curve (*AUC*) [19,70] as evaluation criteria.

4.2. Study on the preprocessing methods

In this section, we analyze the behavior of the preprocessing methods on imbalanced datasets. For this purpose, we compare some of the most representative techniques, previously presented in Section 3.1, developing a ranking according to the

[1] http://www.keel.es/datasets.php.

Table 4
Average AUC results for the preprocessing techniques.

Preprocessing	C4.5		SVM		kNN	
	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}
None	.8790 ±.1226	.7873 ±.1437	.7007 ±.1706	.6891 ±.1681	.8011 ±.1339	.8028 ±.1383
SMOTE	.9613 ±.0504	.8288 ±.1192	.8631 ±.1045	**.8470 ±.1152**	.9345 ±.1247	.8341 ±.1194
SMOTE+ENN	.9482 ±.0525	**.8323 ±.1166**	.8815 ±.1001	.8461 ±.1162	.9284 ±.1262	**.8443 ±.1158**
Border-SMOTE	.9333 ±.0595	.8187 ±.1272	.9082 ±.0941	.8397 ±.1163	.9144 ±.0682	.8177 ±.1314
SL-SMOTE	.9175 ±.0615	.8285 ±.1112	.8365 ±.1020	.8427 ±.1176	.8024 ±.1331	.8029 ±.1381
ADASYN	.9589 ±.0469	.8225 ±.1234	.8283 ±.1054	.8323 ±.1148	.9347 ±.0500	.8355 ±.1163
SPIDER2	.9684 ±.0378	.8018 ±.1329	.7252 ±.1493	.7371 ±.1542	.8381 ±.1176	.8207 ±.1338
DBSMOTE	.8908 ±.1006	.7877 ±.1441	.8612 ±.0778	.7546 ±.1368	.8147 ±.1163	.8082 ±.1293

performance obtained in each case. This representative set of methods is composed by the following techniques: SMOTE [27], SMOTE+ENN [9], Borderline-SMOTE (Border-SMOTE) [63], Adaptive Synthetic Sampling (ADASYN) [65], Safe-Level-SMOTE (SL-SMOTE) [21], SPIDER2 [97] and DBSMOTE [22]. In all cases we try to obtain a level of balance in the training data near to the 50:50 distribution. Additionally, the interpolations that are computed to generate new synthetic data are made considering the 5-nearest neighbors of minority class instances using the euclidean distance.

In Table 4 we show the average results for all preprocessing methods, also including the performance with the original data (None). In bold, we highlight the preprocessing method that obtains the best performing average within each group. We observe that, in all cases, the oversampling mechanisms are very good solutions for achieving a higher performance by comparison to using the original training data.

This behavior is contrasted in Fig. 4, where we have ordered the corresponding methods according to their AUC results in testing for each dataset, considering the average ranking value. We must stress SMOTE+ENN and SMOTE as the top methodologies, since they obtain the highest rank for the three classification algorithms used in this study. We can also observe that both Border-SMOTE and ADASYN are quite robust on average, obtaining a fair average ranking for all datasets.

For the sake of finding out which algorithms are distinctive among an $n \times n$ comparison, we carry out a Shaffer post hoc test [113], which is shown in Tables 5–7. In these tables, a "+" symbol implies that the algorithm in the row is statistically better than the one in the column, whereas "−" implies the contrary; "=" means that the two algorithms compared show no significant differences. In brackets, the adjusted p-value associated to each comparison is shown.

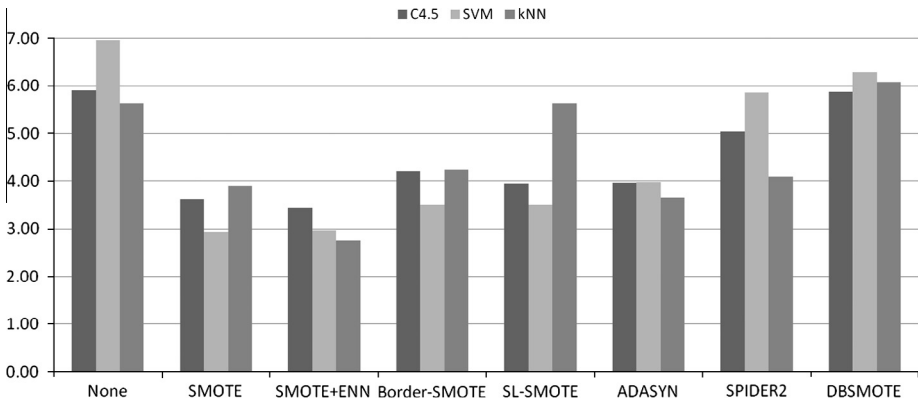


Fig. 4. Average ranking of the preprocessing algorithms for classification with imbalanced datasets.

Table 5
Shaffer test for the preprocessing techniques with C4.5 using the AUC measure.

C4.5	None	SMOTE	SMOTE+ENN	Border-SMOTE	SL-SMOTE	ADASYN	SPIDER2	DBSMOTE
None	x	−(.000002)	−(.000000)	−(.001104)	−(.000096)	−(.000124)	=(.580860)	=(1.00000)
SMOTE	+(.000002)	x	=(1.00000)	=(1.00000)	=(1.00000)	=(1.00000)	+(.013398)	+(.000003)
SMOTE+ENN	+(.000000)	=(1.00000)	x	=(.769498)	=(1.00000)	=(1.00000)	+(.002466)	+(.000000)
Border-SMOTE	+(.001104)	=(1.00000)	=(.769498)	x	=(1.00000)	=(1.00000)	=(.631767)	+(.001379)
SL-SMOTE	+(.000096)	=(1.00000)	=(1.00000)	=(1.00000)	x	=(1.00000)	=(.159840)	+(.000124)
ADASYN	+(.000124)	=(1.00000)	=(1.00000)	=(1.00000)	=(1.00000)	x	=(.174600)	+(.000159)
SPIDER2	=(.580860)	−(.013398)	−(.002466)	=(.631767)	=(.159840)	=(.174600)	x	=(.631767)
DBSMOTE	=(1.00000)	−(.000003)	−(.000000)	−(.001379)	−(.000124)	−(.000159)	=(.631767)	x

Table 6
Shaffer test for the preprocessing techniques with SVM using the AUC measure.

SVM	None	SMOTE	SMOTE+ENN	Border-SMOTE	SL-SMOTE	ADASYN	SPIDER2	DBSMOTE
None	x	−(.000000)	−(.000000)	−(.000000)	−(.000000)	−(.000000)	=(.129870)	=(1.00000)
SMOTE	+(.000000)	x	=(1.00000)	=(1.00000)	=(1.00000)	=(.179175)	+(.000000)	+(.000000)
SMOTE+ENN	+(.000000)	=(1.00000)	x	=(1.00000)	=(1.00000)	=(.199418)	+(.000000)	+(.000000)
Border-SMOTE	+(.000000)	=(1.00000)	=(1.00000)	x	=(1.00000)	=(1.00000)	+(.000000)	+(.000000)
SL-SMOTE	+(.000000)	=(1.00000)	=(1.00000)	=(1.00000)	x	=(1.00000)	+(.000000)	+(.000000)
ADASYN	+(.000000)	=(.179175)	=(.199418)	=(1.00000)	=(1.00000)	x	+(.000126)	+(.000001)
SPIDER2	=(.129870)	−(.000000)	−(.000000)	−(.000000)	−(.000000)	−(.000126)	x	=(1.00000)
DBSMOTE	=(1.00000)	−(.000000)	−(.000000)	−(.000000)	−(.000000)	−(.000001)	=(1.00000)	x

Table 7
Shaffer test for the preprocessing techniques with kNN using the AUC measure.

kNN	None	SMOTE	SMOTE+ENN	Border-SMOTE	SL-SMOTE	ADASYN	SPIDER2	DBSMOTE
None	x	−(.000757)	−(.000000)	−(.014934)	=(1.00000)	−(.000081)	−(.004963)	=(1.00000)
SMOTE	+(.000757)	x	−(.089266)	=(1.00000)	+(.000701)	=(1.00000)	=(1.00000)	+(.000006)
SMOTE+ENN	+(.000000)	+(.089266)	x	+(.007968)	+(.000000)	=(.360402)	+(.022513)	+(.000000)
Border-SMOTE	+(.014934)	=(1.00000)	−(.007968)	x	+(.014027)	=(1.00000)	=(1.00000)	+(.000253)
SL-SMOTE	=(1.00000)	−(.000701)	−(.000000)	−(.014027)	x	−(.000074)	−(.004634)	=(1.00000)
ADASYN	+(.000081)	=(1.00000)	=(.360402)	=(1.00000)	+(.000074)	x	=(1.00000)	+(.000000)
SPIDER2	+(.004963)	=(1.00000)	−(.022513)	=(1.00000)	+(.004634)	=(1.00000)	x	+(.000062)
DBSMOTE	=(1.00000)	−(.000006)	−(.000000)	−(.000253)	=(1.00000)	−(.000000)	−(.000062)	x

In order to explain why SMOTE+ENN and SMOTE obtain the highest performance, we may emphasize two feasible reasons. The first one is related to the addition of significant information within the minority class examples by including new synthetic examples. These new examples allow the formation of larger clusters to help the classifiers to separate both classes, and the cleaning procedure also adds benefits to the generalization ability during learning. The second reason is that the more sophisticated the technique is, the less general it becomes for the high number of benchmark problems selected for our study.

According to these results, we select both SMOTE+ENN and SMOTE as good behaving methodologies for our final comparison.

4.3. Study on the cost-sensitive learning algorithms

In this section, we carry out an analysis regarding cost-sensitive classifiers. We use three different approaches, namely "Weighted-Classifier" (CS-Weighted) [7,123], MetaCost [38], and the CostSensitive Classifier (CS-Classifier) from the Weka environment [62]. In the first case, the base classifiers are modified usually by weighting the instances of the dataset to take into account the a priori probabilities, according to the number of samples in each class. In the two latter cases, we use an input cost-matrix defining $C(+,-) = IR$ and $C(-,+) = 1$.

Table 8 shows the average AUC results where the best average values per algorithm group are highlighted in bold. From this table, we may conclude, as in the previous case for preprocessing, the goodness of the use of this type of solution for imbalanced data, as there is a significant difference with respect to the results obtained with the original data. We may also observe the good behavior of the "CS-Weighted" in contrast with the remaining techniques, and also the good accuracy for the MetaCost algorithm, for both C4.5 and kNN.

Fig. 5 presents the ranking for the selected methods. We can appreciate that the "CS-Weighted" approach achieves the highest rank overall, as pointed out before. The MetaCost method obtains also a good average for C4.5 and kNN, but it is outperformed by the CS-Classifier when SVM is used.

As in the latter case, we show a Shaffer post hoc test for detecting significant differences among the results (Tables 9–11).

Table 8
Average AUC results for the cost-sensitive learning techniques.

Cost-sensitive	C4.5		SVM		kNN	
	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}
None	.8790 ±.1226	.7873 ±.1437	.7007 ±.1706	.6891 ±.1681	.8011 ±.1339	.8028 ±.1383
CS-Weighted	.9711 ±.0580	.8284 ±.1263	.8751 ±.1068	**.8464 ±.1124**	.8427 ±.1201	**.8463 ±.1177**
MetaCost	.9159 ±.0797	**.8370 ±.1287**	.6931 ±.1715	.6802 ±.1696	.9849 ±.0118	.8250 ±.1301
CS-Classifier	.8915 ±.1191	.8116 ±.1387	.8701 ±.1053	.8391 ±.1152	.9993 ±.0046	.8084 ±.1343

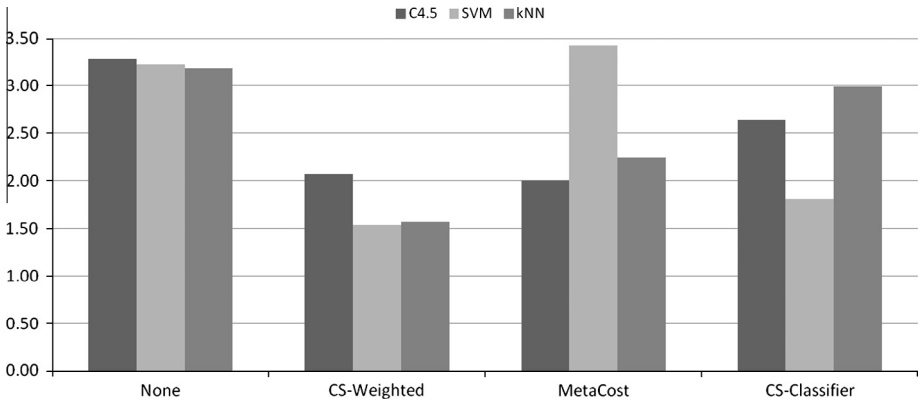


Fig. 5. Average ranking of the cost-sensitive learning algorithms for the classification with imbalanced datasets.

Table 9
Shaffer test for the cost-sensitive learning techniques with C4.5 using the AUC measure.

C4.5	None	CS-Weighted	MetaCost	CS-Classifier
None	x	−(.000000)	−(.000000)	−(.013893)
CS-Weighted	+(.000000)	x	=(.787406)	+(.020817)
MetaCost	+(.000000)	=(.787406)	x	+(.013893)
CS-Classifier	+(.013893)	−(.020817)	−(.013893)	x

Table 10
Shaffer test for the cost-sensitive learning techniques with SVM using the AUC measure.

SVM	None	CS-Weighted	MetaCost	CS-Classifier
None	x	−(.000000)	=(.449832)	−(.000000)
CS-Weighted	+(.000000)	x	+(.000000)	=(.449832)
MetaCost	=(.449832)	−(.000000)	x	−(.000000)
CS-Classifier	+(.000000)	=(.449832)	+(.000000)	x

Table 11
Shaffer test for the cost-sensitive learning techniques with kNN using the AUC measure.

kNN	None	CS-Weighted	MetaCost	CS-Classifier
None	x	−(.000000)	−(.000075)	=(.345231)
CS-Weighted	+(.000000)	x	+(.004828)	+(.000000)
MetaCost	+(.000075)	−(.004828)	x	+(.003228)
CS-Classifier	=(.345231)	−(.000000)	−(.003228)	x

The good behavior shown by introducing weights to the training examples can be explained by its simplicity, because the algorithm procedure is maintained and is adapted to the imbalanced situation. Therefore, it works similarly to an oversampling approach but without adding new samples and complexity to the problem itself. On the other hand, the MetaCost method follows a similar aim, therefore obtaining high quality results. Regarding these facts, we will select these two methods as the representative ones for this family.

4.4. Study on the ensemble-based techniques

The last family of approaches for dealing with imbalanced datasets that we will analyze is the one based on ensemble techniques. In this case, we have selected five different algorithms which showed a very good behavior on the study carried out in [50], namely AdaBoost.M1 (AdaB-M1) [110], AdaBoost with costs outside the exponent (AdaC2) [117], RUSBoost (RUSB) [112], SMOTEBagging (SBAG) [130], and EasyEnsemble (EASY) [85]. We must point out that AdaB-M1 was not included in the taxonomy presented in Section 3.3 since it is not strictly oriented towards imbalanced classification, but we have decided to study it as a classical ensemble approach and because it has shown a good behavior in [50]. Regarding the number of internal classifiers used within each approach, AdaB-M1, AdaC2 and SBAG use 40 classifiers, whereas the remaining approaches use only 10. Additionally, EASY considers 4 bags for the learning stage.

Table 12
Average AUC results for the ensemble methodologies.

Ensemble	C4.5		SVM		kNN	
	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}
None	.8790 ± .1226	.7873 ± .1437	.7007 ± .1706	.6891 ± .1681	.8011 ± .1339	.8028 ± .1383
AdaB-M1	.9915 ± .0468	.8072 ± .1334	.7862 ± .1659	.7615 ± .1630	.9983 ± .0101	.8090 ± .1345
AdaC2	.9470 ± .0858	.8188 ± .1312	.6366 ± .1497	.6271 ± .1479	.9991 ± .0062	.8080 ± .1344
RUSB	.9481 ± .0545	.8519 ± .1129	.7667 ± .1652	.7517 ± .1642	.9359 ± .0495	.8465 ± .1118
SBAG	.9626 ± .0455	**.8545 ± .1111**	.8662 ± .1050	**.8456 ± .1137**	.9825 ± .0253	**.8485 ± .1164**
Easy	.9076 ± .0626	.8399 ± .1091	.8565 ± .1057	.8370 ± .1150	.9093 ± .0667	.8440 ± .1095

In this case, the average AUC results for training and testing are shown in Table 12. The values highlighted in bold correspond to the algorithms that obtain a better performance according to the base classifier. From this table we may conclude the good performance of RUSB, SBAG and EASY. Among them, SBAG stands out for obtaining slightly better results. Anyway, these three algorithms outperform the others considered in this study. The reader might have also noticed that, the great behavior of RUSB is attained using only 10 base classifiers.

This can also be seen from Fig. 6, where we can observe that these three algorithms obtain the first rank positions in almost all cases. It is noticeable that RUSB decreases its results in the case of the SVM algorithm, which can be due to the removal of significant samples for determining the support vectors for the margin classifier in each iteration of the learning.

Tables 13–15 present a Shaffer test, where we can observe, in a nutshell, the statistical differences among the ensemble methodologies selected for this study.

Nevertheless, we must point out that more complex methods do not perform much better than simpler ones. Bagging techniques are easy to develop, but also powerful when dealing with class imbalance if they are properly combined. Their

Table 13
Shaffer test for the ensemble methodologies with C4.5 using the AUC measure.

C4.5	None	AdaB-M	AdaC2	RUSB	SBAG	Easy
None	x	=(.214054)	−(.000767)	−(.000000)	−(.000000)	−(.000001)
AdaB-M	=(.214054)	x	=(.137090)	−(.000001)	−(.000000)	−(.00339)
AdaC2	+(.000767)	=(.137090)	x	−(.006691)	−(.00115)	=(.339838)
RUSB	+(.000000)	+(.000001)	+(.006691)	x	=(.641758)	=(.214054)
SBAG	+(.000000)	+(.000000)	+(.00115)	=(.641758)	x	+(.099451)
Easy	+(.000001)	+(.003390)	=(.339838)	=(.214054)	−(.099451)	x

Table 14
Shaffer test for the ensemble methodologies with SVM using the AUC measure.

SVM	None	AdaB-M	AdaC2	RUSB	SBAG	Easy
None	x	−(.000721)	=(.208828)	−(.015681)	−(.000000)	−(.000000)
AdaB-M	+(.000721)	x	+(.000000)	=(.401501)	−(.000001)	−(.000343)
AdaC2	=(.208828)	−(.000000)	x	−(.000018)	−(.000000)	−(.000000)
RUSB	+(.015681)	=(.401501)	+(.000018)	x	−(.000000)	−(.000007)
SBAG	+(.000000)	+(.000001)	+(.000000)	+(.000000)	x	=(.401501)
Easy	+(.000000)	+(.000343)	+(.000000)	+(.000007)	=(.401501)	x

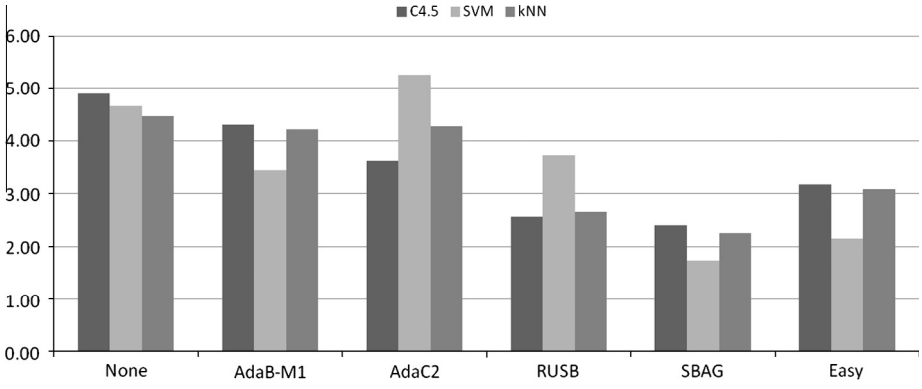


Fig. 6. Average ranking of the ensemble algorithms for the classification with imbalanced datasets.

Table 15
Shaffer test for the ensemble methodologies with kNN using the AUC measure.

kNN	None	AdaB-M	AdaC2	RUSB	SBAG	Easy
None	x	=(1.00000)	=(1.00000)	−(.000000)	−(.000000)	−(.000118)
AdaB-M	=(1.00000)	x	=(1.00000)	−(.000017)	−(.000000)	−(.003106)
AdaC2	=(1.00000)	=(1.00000)	x	−(.000006)	−(.000000)	−(.001517)
RUSB	+(.000000)	+(.000017)	+(.000006)	x	=(.803003)	=(.803003)
SBAG	+(.000000)	+(.000000)	+(.000000)	=(.803003)	x	+(.063015)
Easy	+(.000118)	+(.003106)	+(.001517)	=(.803003)	−(.063015)	x

Table 16
Average global results for C4.5 with the representative methodologies for addressing imbalanced classification.

Preprocessing	C4.5		SVM		kNN	
	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}
None	.8790 ± .1226	.7873 ± .1437	.7007 ± .1706	.6891 ± .1681	.8011 ± .1339	.8028 ± .1383
SMOTE	.9613 ± .0504	.8288 ± .1192	.8631 ± .1045	**.8470 ± .1152**	.9345 ± .1247	.8341 ± .1194
SMOTE+ENN	.9482 ± .0525	.8323 ± .1166	.8815 ± .1001	.8461 ± .1162	.9284 ± .1262	.8443 ± .1158
CS-Weighted	.9711 ± .0580	.8284 ± .1263	.8751 ± .1068	.8464 ± .1124	.8427 ± .1201	.8463 ± .1177
MetaCost	.9159 ± .0797	.8370 ± .1287	.6931 ± .1715	.6802 ± .1696	.9849 ± .0118	.8250 ± .1301
RUSB	.9481 ± .0545	.8519 ± .1129	.7667 ± .1652	.7517 ± .1642	.9359 ± .0495	.8465 ± .1118
SBAG	.9626 ± .0455	**.8545 ± .1111**	.8662 ± .1050	.8456 ± .1137	.9825 ± .0253	**.8485 ± .1164**

hybridization with data preprocessing techniques has shown competitive results and the key issue of these methods resides in properly exploiting the diversity when each bootstrap replica is formed.

Since we have to select only two methodologies for the global analysis, we will stress SBAG as the best ranked method and RUSB, because it presents a robust behavior on average and the second best mean performance in two of the three algorithms.

4.5. Global analysis for the methodologies that address imbalanced classification

In this last section of the experimental analysis on the behavior of the methodologies for addressing classification with imbalanced datasets, we will perform a cross-family comparison for the approaches previously selected as the representatives for each case, namely preprocessing (SMOTE and SMOTE+ENN), cost-sensitive learning (CS-Weighted and MetaCost) and ensemble techniques (RUSB and SBAG). The global results are shown in Table 16, whereas the new performance ranking is shown in Fig. 7. As in the previous cases, the bold values in Table 16 correspond to the algorithms that obtain the highest performance.

Considering these results, we must highlight the dominance of the ensemble approaches versus the remaining models for the "weak classifiers", i.e. C4.5 and kNN. For SVM, the best results are achieved by preprocessing and CS-weighted, showing the significance of adjusting the objective function towards the positive instances, for biasing the separating hyperplane. Regarding the comparison between the cost-sensitive classifiers and the oversampling methods, we observe that, on average, SMOTE+ENN, CS-Weighted and SMOTE obtain very good results and, therefore, they have a similar ranking, followed by the MetaCost method. We must point out that these conclusions regarding the latter techniques are in concordance with the study done in [88].

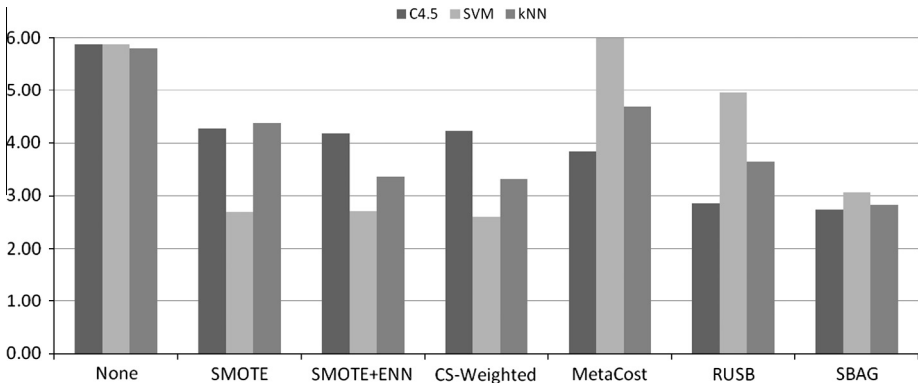


Fig. 7. Average ranking of the representative algorithms for the classification with imbalanced datasets.

Table 17
Shaffer test for the representative methodologies with C4.5 using the AUC measure.

C4.5	None	SMOTE	SMOTE+ENN	CS-Weighted	MetaCost	RUSB	SBAG
None	x	−(.000292)	−(.000087)	−(.000203)	−(.000001)	−(.000000)	−(.000000)
SMOTE	+(.000292)	x	=(1.00000)	=(1.00000)	=(1.00000)	−(.001816)	−(.000648)
SMOTE+ENN	+(.000087)	=(1.00000)	x	=(1.00000)	=(1.00000)	−(.004560)	−(.001423)
CS-Weighted	+(.000203)	=(1.00000)	=(1.00000)	x	=(1.00000)	−(.002500)	−(.000671)
MetaCost	+(.000001)	=(1.00000)	=(1.00000)	=(1.00000)	x	−(.061745)	−(.02942)
RUSB	+(.000000)	+(.001816)	+(.004560)	+(.002500)	+(.061745)	x	=(1.00000)
SBAG	+(.000000)	+(.000648)	+(.001423)	+(.000671)	+(.02942)	=(1.00000)	x

Table 18
Shaffer test for the representative methodologies with SVM using the AUC measure.

SVM	None	SMOTE	SMOTE+ENN	CS-Weighted	MetaCost	RUSB	SBAG
None	x	−(.000000)	−(.000000)	−(.000000)	=(1.00000)	−(.097865)	−(.000000)
SMOTE	+(.000000)	x	=(1.00000)	=(1.00000)	+(.000000)	+(.000000)	=(1.00000)
SMOTE+ENN	+(.000000)	=(1.00000)	x	=(1.00000)	+(.000000)	+(.000000)	=(1.00000)
CS-Weighted	+(.000000)	=(1.00000)	=(1.00000)	x	+(.000000)	+(.000000)	=(1.00000)
MetaCost	=(1.00000)	−(.000000)	−(.000000)	−(.000000)	x	−(.019779)	−(.000000)
RUSB	+(.097865)	−(.000000)	−(.000000)	−(.000000)	+(.019779)	x	−(.000005)
SBAG	+(.000000)	=(1.00000)	=(1.00000)	=(1.00000)	+(.000000)	+(.000005)	x

Table 19
Shaffer test for the representative methodologies with kNN using the AUC measure.

kNN	None	SMOTE	SMOTE+ENN	CS-Weighted	MetaCost	RUSB	SBAG
None	x	−(.002684)	−(.000000)	−(.000000)	−(.038367)	−(.000000)	−(.000000)
SMOTE	+(.002684)	x	−(.058815)	−(.049543)	=(1.00000)	=(.371813)	−(.000545)
SMOTE+ENN	+(.000000)	+(.058815)	x	=(1.00000)	+(.004309)	=(1.00000)	=(.950901)
CS-Weighted	+(.000000)	+(.049543)	=(1.00000)	x	+(.002705)	=(1.00000)	=(.986440)
MetaCost	+(.038367)	=(1.00000)	−(.004309)	−(.002705)	x	−(.057811)	−(.000011)
RUSB	+(.000000)	=(.371813)	=(1.00000)	=(1.00000)	+(.057811)	x	=(.196710)
SBAG	+(.000000)	+(.000545)	=(.950901)	=(.986440)	+(.000011)	=(.196710)	x

In the same way as in the previous sections of this study, we proceed with a Shaffer test (Tables 17–19) that aims to contrast whether two algorithms are significantly different and how different they are.

As a final remark, we must state that all the solutions analyzed here present different particularities, which make them more appropriate for a given application. For example, ensemble methodologies have shown to be very accurate, but their learning time may be high and the output model can be difficult to comprehend by the final user. Cost-sensitive approaches have also shown to be very precise, but the necessity of defining an optimal cost-matrix impose hard restrictions to their use. Finally, the preprocessing algorithms have shown their robustness and obtained very good global results, and therefore they can be viewed as a standard approach for imbalanced datasets.

5. Problems related to data intrinsic characteristics in imbalanced classification

As it was stated in the introduction of this work, skewed class distributions do not hinder the learning task by itself [66,118], but usually a series of difficulties related with this problem turn up. This issue is depicted in Fig. 8, in which we show the performance of the SBAG with the different datasets used in the previous section, ordered according to the IR, in order to search for some regions of interesting good or bad behavior. As we can observe, there is no pattern of behavior for any range of IR, and the results can be poor both for low and high imbalanced data.

Related to this issue, in this section we aim to make a discussion on the nature of the problem itself, emphasizing several data intrinsic characteristics that do have a strong influence on imbalanced classification, in order to be able to address this problem in a more feasible way.

With this objective in mind, we focus our analysis on using the C4.5 classifier, in order to develop a basic but descriptive study by showing a series of patterns of behavior, following a kind of "educational scheme". With respect to the previous section, which was carried out in an empirical way, this part of the study is devoted to enumerating the scenarios that can be found when dealing with classification with imbalanced data, emphasizing their main issues that will allow us to design a better algorithm that can be adapted to different niches of the problem.

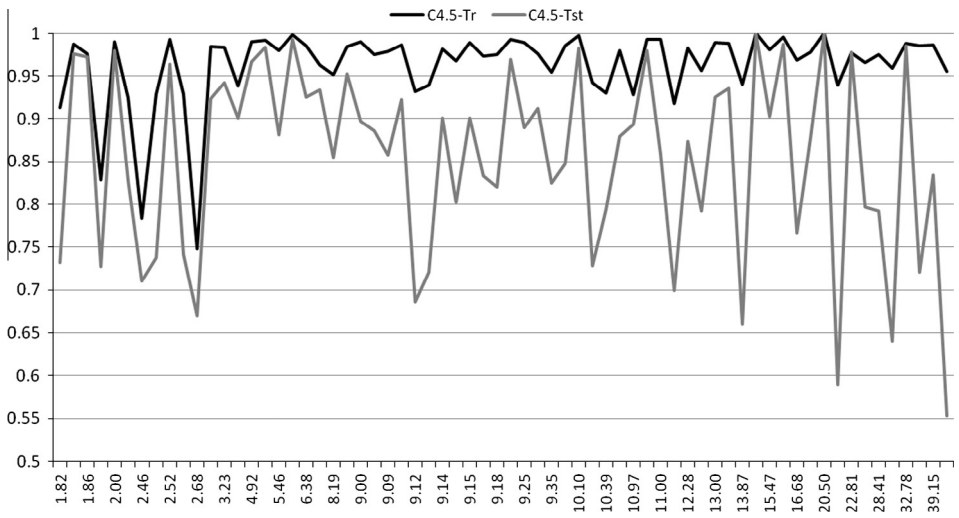


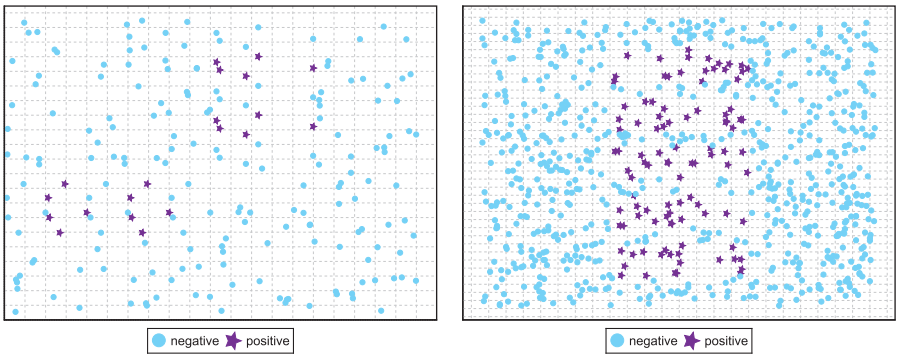
Fig. 8. Performance in training and testing for the C4.5 decision tree with SBAG as a function of IR.

We acknowledge that some of the data intrinsic characteristics described along this section share some features and it is usual that, for a given dataset, several "sub-problems" can be found simultaneously. Nevertheless, we consider a simplified view of all these scenarios to serve as a global introduction to the topic.

First, we discuss about the difficulties related to the presence of small disjuncts in the imbalanced data (Section 5.1). Then, we present the issues about the size of the dataset and the lack of density in the training set (Section 5.2). Next, we focus on the class overlap, showing that it is extremely significant on imbalanced domains (Section 5.3). Then, we analyze the presence of noisy data in this type of problems and how it affects the behavior of both preprocessing techniques and classification algorithms (Section 5.4). After that, we introduce the concept of borderline instances and its relationship with noise examples (Section 5.5). Finally, we define the dataset shift problem in the classification with imbalanced datasets (Section 5.6).

5.1. Small disjuncts

The presence of the imbalanced classes is closely related to the problem of small disjuncts. This situation occurs when the concepts are represented within small clusters, which arise as a direct result of underrepresented subconcepts [99,138]. Although those small disjuncts are implicit in most of the problems, the existence of this type of areas highly increases the complexity of the problem in the case of class imbalance, because it becomes hard to know whether these examples represent an actual subconcept or are merely attributed to noise [73]. This situation is represented in Fig. 9, where we show an artificially generated dataset with small disjuncts for the minority class and the *"Subclus"* problem created in [97], where we can find small disjuncts for both classes: the negative samples are underrepresented with respect to the positive samples in the central region of positive rectangular areas, while the positive samples only cover a small part of the whole dataset and are placed inside the negative class. We must point out that, in all figures of this section, positive instances are represented with dark stars whereas negative instances are depicted with light circles.



(a) Artificial dataset: small disjuncts for the minority class

(b) Subclus dataset: small disjuncts for both classes

Fig. 9. Example of small disjuncts on imbalanced data.

The problem of small disjuncts becomes accentuated for those classification algorithms which are based on a divide-and-conquer approach [135]. This methodology consists in subdividing the original problem into smaller ones, such as the procedure used in decision trees, and can lead to data fragmentation [49], that is, to obtain several partitions of data with a few representation of instances. If the IR of the data is high, this handicap is obviously more severe.

Several studies by Weiss [136,137] analyze this factor in depth and enumerate several techniques for handling the problem of small disjuncts:

1. **Obtain additional training data**. The lack of data can induce the apparition of small disjuncts, especially in the minority class, and these areas may be better covered just by employing an informed sampling scheme [71].
2. **Use a more appropriate inductive bias**. If we aim to be able to properly detect the areas of small disjuncts, some sophisticated mechanisms must be employed for avoiding the preference for the large areas of the problem. For example, [68] modified CN2 so that its maximum generality bias is used only for large disjuncts, and a maximum specificity bias was then used for small disjuncts. However, this approach also degrades the performance of the small disjuncts, and some authors proposed to refine the search and to use different learners for the examples that fall in the large disjuncts and on the small disjuncts separately [24,121].
3. **Using more appropriate metrics**. This issue is related to the previous one in the sense that, for the data mining process, it is recommended to use specific measures for imbalanced data, in a way that the minority classes in the small disjuncts are positively weighted when obtaining the classification model [134]. For example, the use of precision and recall for the minority and majority classes, respectively, can lead to generate more precise rules for the positive class [41,74].
4. **Disabling pruning**. Pruning tends to eliminate most small disjuncts by a generalization of the obtained rules. Therefore, this methodology is not recommended.
5. **Employ boosting**. Boosting algorithms, such as the AdaBoost algorithm, are iterative algorithms that place different weights on the training distribution each iteration [110]. Following each iteration, boosting increases the weights associated with the incorrectly classified examples and decreases the weights associated with the correctly classified examples. Because instances in the small disjuncts are known to be difficult to predict, it is reasonable to believe that boosting will improve their classification performance. Following this idea, many approaches have been developed by modifying the standard boosting weight-update mechanism in order to improve the performance on the minority class and the small disjuncts [30,44,61,69,74,112,117,122].

Finally, we must emphasize the use of the CBO method [73], which is a resampling strategy that is used to counteract simultaneously the between-class imbalance and the within-class imbalance. Specifically, this approach detects the clusters in the positive and negative classes using the *k*-means algorithm in a first step. In a second step, it randomly replicates the examples for each cluster (except the largest negative cluster) in order to obtain a balanced distribution between clusters from the same class and between classes. These clusters can be viewed as small disjuncts in the data, and therefore this preprocessing mechanism is aimed to stress the significance of these regions.

In order to show the goodness of this approach, we depict a short analysis on the two previously presented artificial datasets, that is, our artificial problem and the Subclus dataset, studying the behavior of the C4.5 classifier according to both the differences in performance between the original and the preprocessed data and the boundaries obtained in each case. We must point out that the whole dataset is used in both cases.

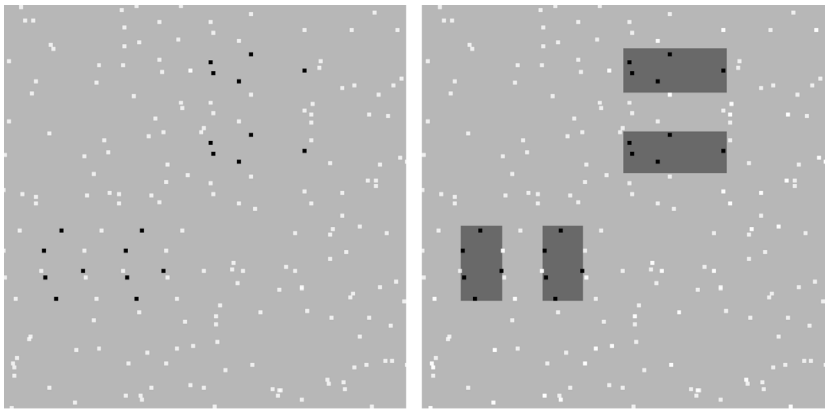
Table 20 shows the results of C4.5 in each case, where we must emphasize that the application of CBO enables the correct identification of all the examples for both classes. Regarding the visual output of the C4.5 classifier (Fig. 10), in the first case we observe that for the original data no instances of the positive class are recognized, and that there is an overgeneralization of the negative instances, whereas the CBO method achieves the correct identification of the four clusters in the data, by replicating an average of 11.5 positive examples and 1.25 negative examples. In the Subclus problem, there is also an overgeneralization for the original training data, but in this case we found that the small disjuncts of the negative class surrounding the positive instances are the ones which are misclassified now. Again, the application of the CBO approach results on a perfect classification for all data, having 7.8 positive instances for each "data point" and 1.12 negative ones.

5.2. Lack of density

One problem that can arise in classification is the small sample size [106]. This issue is related to the "lack of density" or "lack of information", where induction algorithms do not have enough data to make generalizations about the distribution of

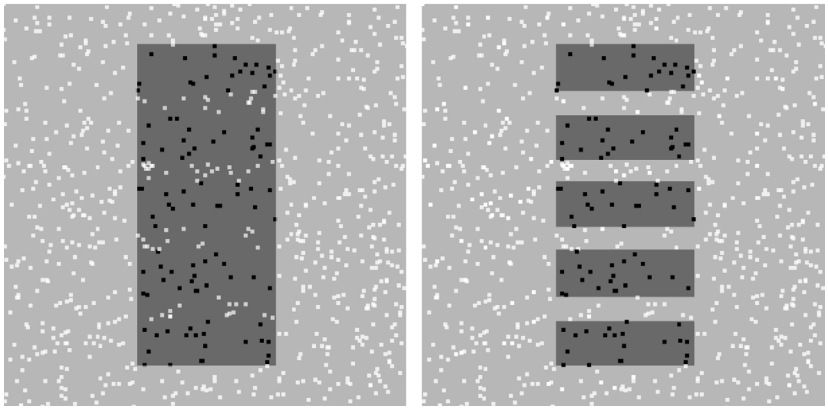
Table 20
Performance obtained by C4.5 in datasets suffering from small disjuncts.

Dataset	Original data			Preprocessed data with CBO		
	TP_{rate}	TN_{rate}	AUC	TP_{rate}	TN_{rate}	AUC
Artificial dataset	.0000	1.000	.5000	1.000	1.000	1.000
Subclus dataset	1.000	.9029	.9514	1.000	1.000	1.000



(a) Artificial dataset with the original data: 20 positive and 182 negative instances

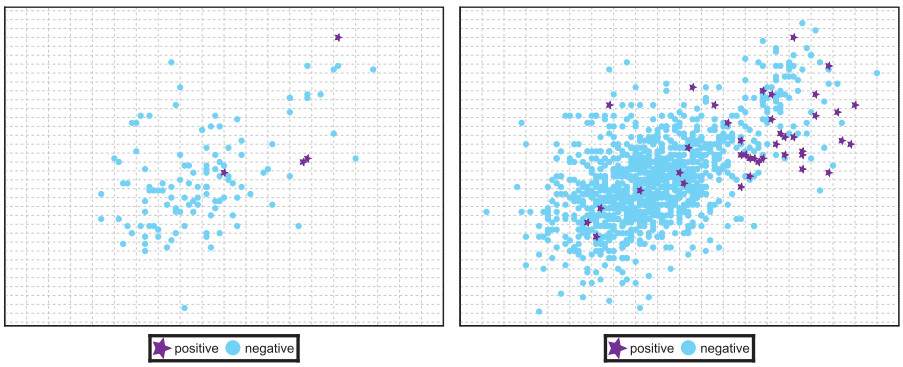
(b) Artificial dataset with CBO: 228 positive and 228 negative instances



(c) Subclus dataset with the original data: 100 positive and 700 negative instances

(d) Subclus dataset with CBO: 780 positive and 780 negative instances

Fig. 10. Boundaries obtained by C4.5 with the original and preprocessed data using CBO for addressing the problem of small disjuncts. The new instances for (b) and (d) are just replicates of the initial examples.



(a) 10 % of training instances

(b) 100 % of training instances

Fig. 11. Lack of density or small sample size on the yeast4 dataset.

samples, a situation that becomes more difficult in the presence of high dimensional and imbalanced data. A visual representation of this problem is depicted in Fig. 11, where we show a scatter plot for the training data of the yeast4 problem (attributes mcg vs. gvh) only with a 10% of the original instances (Fig. 11a) and and with all the data (Fig. 11b). We can appreciate that it becomes very hard for the learning algorithm to obtain a model that is able to perform a good generalization when there is not enough data that represents the boundaries of the problem and, what it is also most significant, when the concentration of minority examples is so low that they can be simply treated as noise.

The combination of imbalanced data and the small sample size problem presents a new challenge to the research community [133]. In this scenario, the minority class can be poorly represented and the knowledge model to learn this data space becomes too specific, leading to overfitting. Furthermore, as stated in the previous section, the lack of density in the training data may also cause the introduction of small disjuncts. Therefore, two datasets cannot be considered to present the same complexity because they have the same IR, as it is also important how the training data represents the minority instances.

In [138], the authors have studied the effect of class distribution and training-set size on the classifier performance using C4.5 as base learning algorithm. Their analysis consisted in varying both the available training data and the degree of imbalance for several datasets and observing the differences for the AUC metric in those cases.

The first finding they extracted is somehow quite trivial, that is, the higher the number of training data, the better the performance results are, independently of the class distribution. A second important fact that they highlighted is that the IR that yields the best performances occasionally vary from one training-set size to another, giving the support to the notion that there may be a "best" marginal class distribution for a learning task and suggests that a progressive sampling algorithm may be useful in locating the class distribution that yields the best, or nearly best, classifier performance.

In order to visualize the effect of the density of examples in the learning process, in Fig. 12 we show the results in AUC for the C4.5 classifier both for training (black line) and testing (grey line) for the *vowel0* problem, varying the percentage of training instances from 10% to the original training size. This short experiment is carried out on a 5-fold cross-validation, where the test data is not modified, i.e. in all cases it represents a 20% of the original data; the results shown are the average of the five partitions.

From this graph, we may distinguish a growth rate directly proportional to the number of training instances that are being used. This behavior reflects the findings enumerated previously from [138].

5.3. Overlapping or class separability

The problem of overlapping between classes appears when a region of the data space contains a similar quantity of training data from each class. This situation leads to develop an inference with almost the same a priori probabilities in this overlapping area, which makes very hard or even impossible the distinction between the two classes. Indeed, any "linearly separable" problem can be solved by any simple classifier regardless of the class distribution.

There are several works which aim to study the relationship between overlapping and class imbalance. Particularly, in [102] one can find a study where the authors propose several experiments with synthetic datasets varying the imbalance ratio and the overlap existing between the two classes. Their conclusions stated that the class probabilities are not the main responsibles for the hinder in the classification performance, but instead the degree of overlapping between the classes.

To reproduce the example for this scenario, we have created an artificial dataset with 1,000 examples having an IR of 9, i.e. 1 positive instance per 10 instances. Then, we have varied the degree of overlap for individual feature values, from no overlap to 100% of overlap, and we have used the C4.5 classifier in order to determine the influence of overlapping with respect to a fixed IR. First, Table 21 shows the results for the considered cases, where we observe that the performance is highly degrading with the increase of the overlap. Additionally, Fig. 13 shows this issue, where we can observe that the decision tree is not only unable to obtain a correct discrimination between both classes when they are overlapped, but also that the preferred class is the majority one, leading to low values for the AUC metric.

Additionally, in [55], a similar study with several algorithms in different situations of imbalance and overlap focusing on the the kNN algorithm was developed. In this case, the authors proposed two different frameworks: on the one hand, they try to find the relation when the imbalance ratio in the overlap region is similar to the overall imbalance ratio whereas, on the other hand, they search for the relation when the imbalance ratio in the overlap region is inverse to the overall one (the positive class is locally denser than the negative class in the overlap region). They showed that when the overlapped data is not balanced, the IR in overlapping can be more important than the overlapping size. In addition, classifiers using a more global learning procedure attain greater TP rates whereas more local learning models obtain better TN rates than the former.

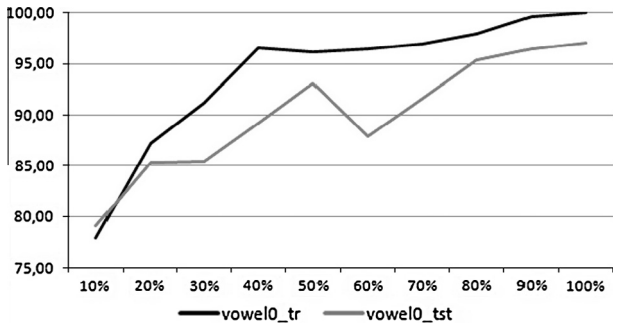
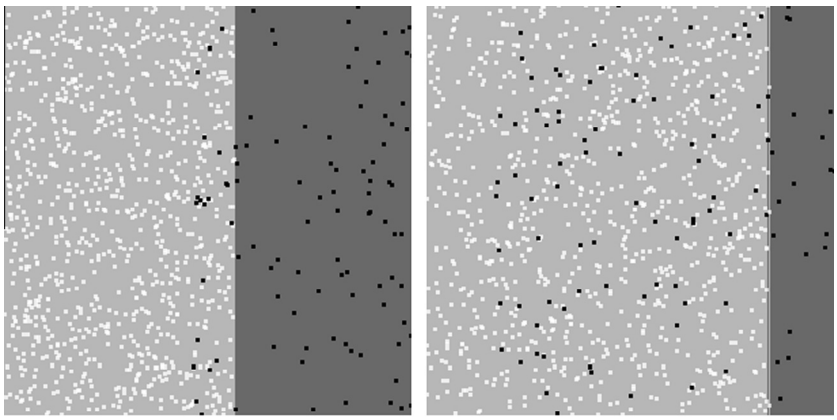


Fig. 12. AUC performance for the C4.5 classifier with respect to the proportion of examples in the training set for the vowel0 problem.

Table 21
Performance obtained by C4.5 with different degrees of overlapping.

Overlap degree (%)	TP_{rate}	TN_{rate}	AUC
0	1.000	1.000	1.000
20	.79.00	1.000	.8950
40	.4900	1.000	.7450
50	.4700	1.000	.7350
60	.4200	1.000	.7100
80	.2100	.9989	.6044
100	.0000	1.000	.5000



(a) 20% overlap (b) 80% overlap

Fig. 13. Example of overlapping imbalanced datasets: boundaries detected by C4.5.

In [37], the authors examine the effects of overlap and imbalance on the complexity of the learned model and demonstrate that overlapping is a far more serious factor than imbalance in this respect. They demonstrate that these two problems acting in concert cause difficulties that are more severe than one would expect by examining their effects in isolation. In order to do so, they also use synthetic datasets for classifying with a SVM, where they vary the imbalance ratio, the overlap between classes and the imbalance ratio and overlap jointly. Their results show that, when the training set size is small, high levels of imbalance cause a dramatic drop in classifier performance, explained by the presence of small disjuncts. Overlapping classes cause a consistent drop in performance regardless of the size of the training set. However, with overlapping and imbalance combined, the classifier performance is degraded significantly beyond what the model predicts.

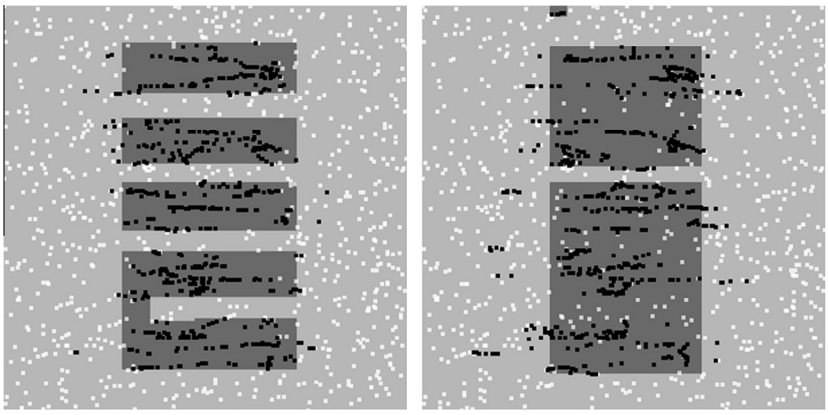
In one of the latest researches on the topic [89], the authors have empirically extracted some interesting findings on real world datasets. Specifically, the authors depicted the performance of the different datasets ordered according to different data complexity measures (including the IR) in order to search for some regions of interesting good or bad behavior. They could not characterize any interesting behavior related to IR, but they do for other metrics that measure the overlap between the classes.

Finally, in [90], an approach that combines preprocessing and feature selection (strictly in this order) is proposed. This approach works in a way where preprocessing deals with class distribution and small disjuncts and feature selection somehow reduces the degree of overlapping. In a more general way, the idea behind this approach tries to overcome different sources of data complexity such as the class overlap, irrelevant and redundant features, noisy samples, class imbalance, low ratios of the sample size to dimensionality and so on, using different approaches used to solve each complexity.

5.4. Noisy data

Noisy data is known to affect the way any data mining system behaves [20,109,151]. Focusing on the scenario of imbalanced data, the presence of noise has a greater impact on the minority classes than on usual cases [135]; since the positive class has fewer examples to begin with, it will take fewer "noisy" examples to impact the learned subconcept. This issue is depicted in Fig. 14, in which we can observe the decision boundaries obtained with SMOTE+C4.5 in the Subclus problem without noisy data (Fig. 14a) and how the frontiers between the classes are wrongly generated by introducing a 20% gaussian noise (Fig. 14b).

According to [135], these "noise-areas" can be somehow viewed as "small disjuncts" and in order to avoid the erroneous generation of discrimination functions for these examples, some overfitting management techniques must be employed,



(a) Original problem and decision functions

(b) Noisy instances and new undesirable decision functions

Fig. 14. Example of the effect of noise in imbalanced datasets for SMOTE+C4.5 in the Subclus dataset.

such as pruning. However, the handicap of this methodology is that some correct minority classes will be ignored and, in this manner, the bias of the learner should be tuned-up in order to be able to provide a good global behavior for both classes of the problem.

For example, Batuwita and Palade developed the FSVM-CIL algorithm [13], a synergy between SVMs and fuzzy logic aimed to reflect the within-class importance of different training examples in order to suppress the effect of outliers and noise. The idea is to assign different fuzzy membership values to positive and negative examples and to incorporate this information in the SVM learning algorithm, aimed to reduce the effect of outliers and noise when finding the separating hyperplane.

In [111] we may find an empirical study on the effect of class imbalance and class noise on different classification algorithms and data sampling techniques. From this study, the authors extracted three important lessons on the topic:

1. Classification algorithms are more sensitive to noise than imbalance. However, as imbalance increases in severity, it plays a larger role in the performance of classifiers and sampling techniques.
2. Regarding the preprocessing mechanisms, simple undersampling techniques such as random undersampling and ENN performed the best overall, at all levels of noise and imbalance. Peculiarly, as the level of imbalance is increased, ENN proves to be more robust in the presence of noise. Additionally, OSS consistently proves itself to be relatively unaffected by an increase in the noise level. Other techniques such as random oversampling, SMOTE or Borderline-SMOTE obtain good results on average, but do not show the same behavior as undersampling.
3. Finally, the most robust classifiers tested over imbalanced and noisy data are bayesian classifiers and SVMs, performing better on average than rule induction algorithms or instance based learning. Furthermore, whereas most algorithms only experience small changes in AUC when imbalance was increased, the performance of Radial Basis Functions is significantly hindered when the imbalance ratio increases. For rule learning algorithms, the presence of noise degrades the performance more quickly than in other algorithms.

Additionally, in [75], the authors presented a similar study on the significance of noise and imbalance data using bagging and boosting techniques. Their results show the goodness of the bagging approach without replacement, and they recommend the use of noise reduction techniques prior to the application of boosting procedures.

As a final remark, we show a brief experimental study on the effect of noise over a specific imbalanced problem such as the Subclus dataset [97]. Table 22 includes the results for C4.5 with no preprocessing (None) and four different approaches, namely random undersampling, SMOTE [27], SMOTE+ENN [9] and SPIDER2 [97], a method designed for addressing noise and borderline examples, which will be detailed in the next section.

This table is divided into two parts, the leftmost columns show the results with the original data and the columns in the right side show the results when adding a 20% of gaussian noise to the data. From this table we may conclude that in all cases the presence of noise degrades the performance of the classifier especially on the positive instances (TP_{rate}). Regarding the preprocessing approaches, the best behavior is obtained by SMOTE+ENN and SPIDER2, both of which include a cleaning mechanism to alleviate the problem of noisy data, whereas the latter also oversample the borderline minority examples.

5.5. Borderline examples

Inspired by [76], we may distinguish between safe, noisy and borderline examples. Safe examples are placed in relatively homogeneous areas with respect to the class label. By noisy examples we understand individuals from one class occurring in safe areas of the other class, as introduced in the previous section. Finally, *Borderline examples* are located in the area

Table 22
Performance obtained by C4.5 in the Subclus dataset with and without noisy instances.

Dataset	Original data			20% of Gaussian noise		
	TP_{rate}	TN_{rate}	AUC	TP_{rate}	TN_{rate}	AUC
None	1.000	.9029	.9514	.0000	1.000	.5000
RandomUnderSampling	1.000	.7800	.8900	.9700	.7400	.8550
SMOTE	.9614	.9529	.9571	.8914	.8800	.8857
SMOTE+ENN	.9676	.9623	.9649	.9625	.9573	.9599
SPIDER2	1.000	1.000	1.000	.9480	.9033	.9256

surrounding class boundaries, where the minority and majority classes overlap. Fig. 15 represents two examples given by [97], named *"Paw"* and *"Clover"*, respectively. In the former, the minority class is decomposed into 3 elliptic subregions, where two of them are located close to each other, and the remaining smaller sub-region is separated (upper right cluster). The latter also represents a non-linear setting, where the minority class resembles a flower with elliptic petals, which makes difficult to determine the boundaries examples in order to carry out a correct discrimination of the classes.

The problem of noisy data and the management of borderline examples are closely related, and most of the cleaning techniques briefly introduced in Section 3.1 can be used, or are the basis for detecting and emphasizing these borderline instances and, what is most important, to distinguish them from noisy instances that can degrade the overall classification. In brief, the better the definition of the borderline areas the more precise the discrimination between the positive and negative classes will be [39].

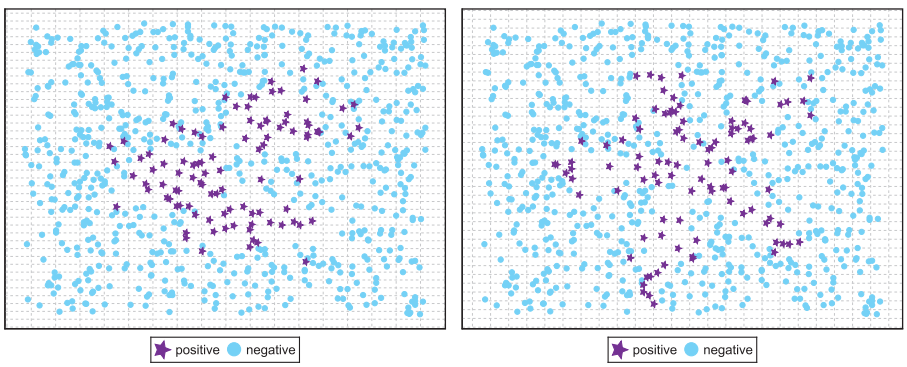
The family of SPIDER methods were proposed in [115] to ease the problem of the improvement of sensitivity at the cost of specificity that appears in the standard cleaning techniques. The SPIDER techniques works by combining a cleaning step of the majority examples with a local oversampling of the borderline minority examples [97,115,116].

We may also find other related techniques such as the Borderline-SMOTE [63], which seeks to oversample the minority class instances in the borderline areas, by defining a set of "Danger" examples, i.e. those which are most likely to be misclassified since they appear in the borderline areas, from which SMOTE generates synthetic minority samples in the neighborhood of the boundaries.

Other approaches such as Safe-Level-SMOTE [21] and ADASYN [65] work in a similar way. The former is based on the premise that previous approaches, such as SMOTE and Borderline-SMOTE, may generate synthetic instances in unsuitable locations, such as overlapping regions and noise regions; therefore, the authors compute a "safe-level" value for each positive instance before generating synthetic instances and generate them closer to the largest safe level. On the other hand, the key idea of the ADASYN algorithm is to use a density distribution as a criterion to automatically decide the number of synthetic samples that need to be generated for each minority example, by adaptively changing the weights of different minority examples to compensate the skewed distributions.

In [87], the authors use a hierarchical fuzzy rule learning approach, which defines a higher granularity for those problem subspaces in the borderline areas. The results have shown to be very competitive for highly imbalanced datasets in which this problem is accentuated.

Finally, in [97], the authors presented a series of experiments in which it is shown that the degradation in performance of a classifier is strongly affected by the number of borderline examples. They showed that focused resampling mechanisms (such as the Neighborhood Cleaning Rule [79] or SPIDER2 [97]) work well when the number of borderline examples is large enough whereas, on the contrary case, oversampling methods allow the improvement of the precision for the minority class.



(a) Paw dataset (b) Clover dataset

Fig. 15. Example of data with difficult borderline examples.

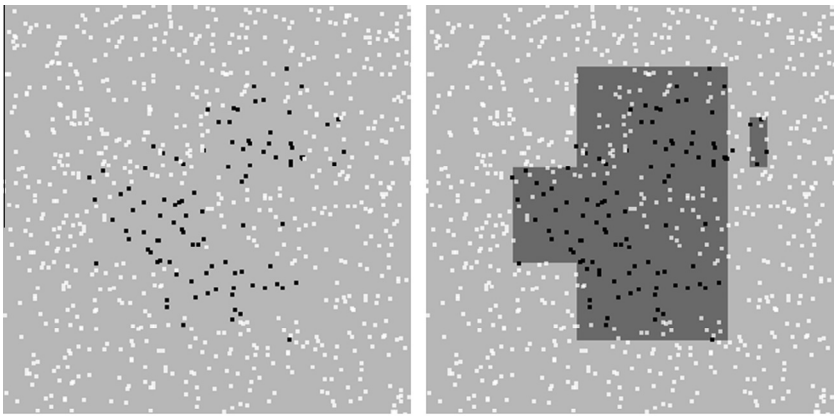


Fig. 16. Boundaries detected by C4.5 in the Paw problem (800 examples, IR 7 and disturbance ratio of 30).

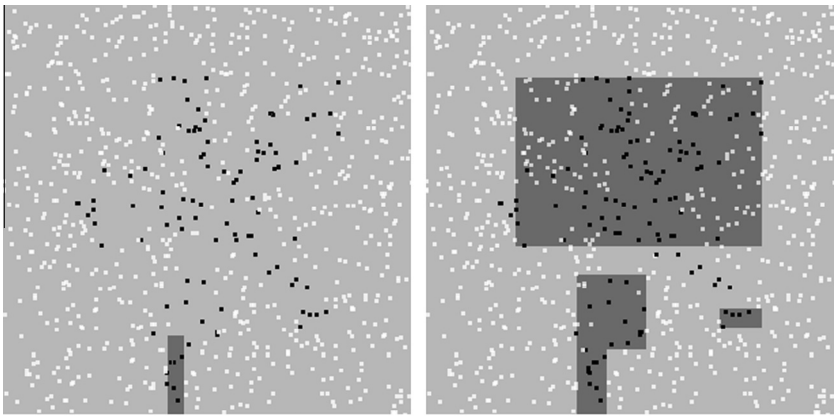


Fig. 17. Boundaries detected by C4.5 in the Clover problem (800 examples, IR 7 and disturbance ratio of 30).

The behavior of the SPIDER2 approach is shown in Table 15 for both the Paw and Clover problems. There are 10 different problems for each one of these datasets, depending on the number of examples and IR (600-5 or 800-7), and the "disturbance ratio" [97], defined as the ratio of borderline examples from the minority class subregions (0–70%). From these results we must stress the goodness of the SPIDER2 preprocessing step especially for those problems with a high disturbance ratio, which are harder to solve.

Additionally, and as a visual example of the behavior of this kind of methods, we show in Figs. 16 and 17 the classification regions detected with C4.5 for the Paw and Clover problems using the original data and applying the SPIDER2 method. From these results we may conclude that the use of a methodology for stressing the borderline areas is very beneficial for correctly identifying the minority class instances (see Table 23).

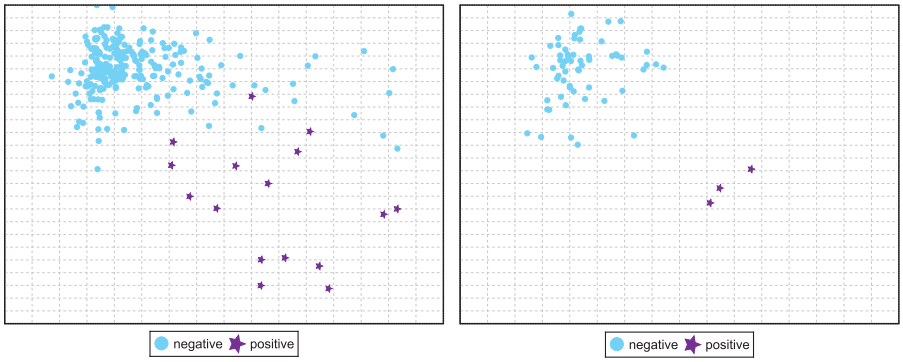
5.6. Dataset shift

The problem of dataset shift [2,23,114] is defined as the case where training and test data follow different distributions. This is a common problem that can affect all kind of classification problems, and it often appears due to sample selection bias issues. A mild degree of dataset shift is present in most real-world problems, but general classifiers are often capable of handling it without a severe performance loss.

However, the dataset shift issue is specially relevant when dealing with imbalanced classification, because in highly imbalanced domains, the minority class is particularly sensitive to singular classification errors, due to the typically low number of examples it presents [94]. In the most extreme cases, a single misclassified example of the minority class can create a significant drop in performance.

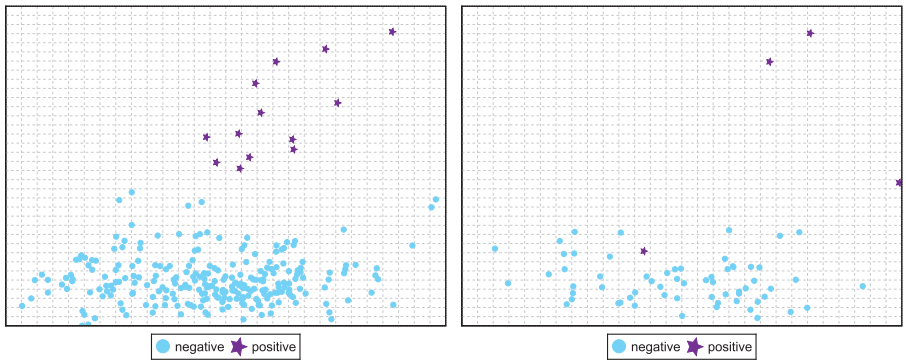
Table 23
AUC results in training and testing for the Clover and Paw problems with C4.5 (Original data and data preprocessed with SPIDER2).

Dataset	Disturbance	600 examples – IR 5				800 examples – IR 7			
		None		SPIDER2		None		SPIDER2	
		AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}	AUC_{Tr}	AUC_{Tst}
Paw	0	.9568	.9100	.9418	.9180	.7095	.6829	.9645	.9457
	30	.7298	.7000	.9150	.8260	.6091	.5671	.9016	.8207
	50	.7252	.6790	.9055	.8580	.5000	.5000	.9114	.8400
	60	.5640	.5410	.9073	.8150	.5477	.5300	.8954	.7829
	70	.6250	.5770	.8855	.8350	.5000	.5000	.8846	.8164
	Average	.7202	.6814	.9110	.8504	.5732	.5560	.9115	.8411
Clover	0	.7853	.7050	.7950	.7410	.7607	.7071	.8029	.7864
	30	.6153	.5430	.9035	.8290	.5546	.5321	.8948	.7979
	50	.5430	.5160	.8980	.8070	.5000	.5000	.8823	.7907
	60	.5662	.5650	.8798	.8100	.5000	.5000	.8848	.8014
	70	.5000	.5000	.8788	.7690	.5250	.5157	.8787	.7557
	Average	.6020	.5658	.8710	.7912	.5681	.5510	.8687	.7864



(a) Training data. AUC = .9043 (b) Test data. AUC = 1.000

Fig. 18. Example of good behavior (no dataset shift) in imbalanced domains: ecoli4 dataset, 5th partition.



(a) Training data. AUC = 1.000 (b) Test data. AUC = .8750

Fig. 19. Example of bad behavior caused by dataset shift in imbalanced domains: ecoli4 dataset, 1st partition.

For clarity, Figs. 18 and 19 present two examples of the influence of the dataset shift in imbalanced classification. In the first case (Fig. 18), it is easy to see a separation between classes in the training set that carries over perfectly to the test set. However, in the second case (Fig. 19), it must be noted how some minority class examples in the test set are at the bottom and rightmost areas while they are localized in other areas in the training set, leading to a gap between the training and testing performance. These problems are represented in a two-dimensional space by means of a linear transformation of the inputs variables, following the technique given in [94].

Since the dataset shift is a highly relevant issue in imbalanced classification, it is easy to see why it would be an interesting perspective to focus on in future research regarding this topic. There are two different potential approaches in the study of the dataset shift in imbalanced domains:

1. The first one focuses on intrinsic dataset shift, that is, the data of interest includes some degree of shift that is producing a relevant drop in performance. In this case, we may develop techniques to discover and measure the presence of dataset shift [32,33,144], but adapting them to focus on the minority class. Furthermore, we may design algorithms that are capable of working under dataset shift conditions, either by means of preprocessing techniques [95] or with ad hoc algorithms [1,16,60]. In both cases, we are not aware of any proposals in the literature that focus on the problem of imbalanced classification in the presence of dataset shift.
2. The second approach in terms of dataset shift in imbalanced classification is related to induced dataset shift. Most current state of the art research is validated through stratified cross-validation techniques, which are another potential source of shift in the learning process. A more suitable validation technique needs to be developed in order to avoid introducing dataset shift issues artificially.

6. Concluding remarks

In this paper, we have reviewed the topic of classification with imbalanced datasets, and focused on two main issues: (1) to present the main approaches for dealing with this problem, namely, preprocessing of instances, cost-sensitive learning and ensemble techniques, and (2) to develop a thorough discussion on the effect of data intrinsic characteristics in learning from imbalanced datasets.

Mainly, we have pointed out that the imbalanced ratio by itself does not have the most significant effect on the classifiers' performance, but that there are other issues that must be taken into account. We have presented six different cases, which, in conjunction with a skewed data distribution, impose a strong handicap for achieving a high classification performance for both classes of the problem, i.e., the presence of small disjuncts, the lack of density or small sample size, the class overlapping, the noisy data, the correct management of borderline examples, and the dataset shift.

For each one of the mentioned issues, we have described the main features that makes learning algorithms to be wrongly biased and we have presented several solutions proposed along the years in the specialized literature. This review paper emphasizes that there is a current need to study the aforementioned intrinsic characteristics of the data, so that future research on classification with imbalanced data should focus on detecting and measuring the most significant data properties, in order to be able to define good solutions as well as alternatives to overcome the problems.

Acknowledgement

This work was partially supported by the Spanish Ministry of Science and Technology under the Project TIN2011-28488 and the Andalusian Research Plans P11-TIC-7765 and P10-TIC-6858. V. López holds a FPU scholarship from the Spanish Ministry of Education.

References

[1] R. Alaiz-Rodríguez, A. Guerrero-Curieses, J. Cid-Sueiro, Improving classification under changes in class and within-class distributions, in: Proceedings of the 10th International Work-Conference on Artificial Neural Networks (IWANN '09), Springer-Verlag, Berlin, Heidelberg, 2009, pp. 122–130.
[2] R. Alaiz-Rodríguez, N. Japkowicz, Assessing the impact of changing environments on classifier performance, in: Proceedings of the 21st Canadian Conference on Advances in Artificial Intelligence (CCAI'08), Springer-Verlag, Berlin, Heidelberg, 2008, pp. 13–24.
[3] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data–mining software tool: data set repository, integration of algorithms and experimental analysis framework, Journal of Multi-Valued Logic and Soft Computing 17 (2–3) (2011) 255–287.
[4] J. Alcalá;-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Computing 13 (2009) 307–318.
[5] A. Anand, G. Pugalenthi, G.B. Fogel, P.N. Suganthan, An approach for classification of highly imbalanced data using weighting and undersampling, Amino Acids 39 (5) (2010) 1385–1391.
[6] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison Wesley, 1999.
[7] R. Barandela, J.S. Sánchez, V. García, E. Rangel, Strategies for learning in class imbalance problems, Pattern Recognition 36 (3) (2003) 849–851.
[8] R. Barandela, R.M. Valdovinos, J.S. Sánchez, New applications of ensembles of classifiers, Pattern Analysis Applications 6 (3) (2003) 245–256.
[9] G.E.A.P.A. Batista, R.C. Prati, M.C. Monard, A study of the behaviour of several methods for balancing machine learning training data, SIGKDD Explorations 6 (1) (2004) 20–29.
[10] R. Batuwita, V. Palade, AGm: a new performance measure for class imbalance learning. application to bioinformatics problems, in: Proceedings of the 8th International Conference on Machine Learning and Applications (ICMLA 2009), 2009, pp. 545–550.
[11] R. Batuwita, V. Palade, microPred: effective classification of pre-miRNAs for human miRNA gene prediction, Bioinformatics 25 (8) (2009) 989–995.
[12] R. Batuwita, V. Palade, Efficient resampling methods for training support vector machines with imbalanced datasets, in: Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN), 2010.
[13] R. Batuwita, V. Palade, FSVM-CIL: fuzzy support vector machines for class imbalance learning, IEEE Transactions on Fuzzy Systems 18 (3) (2010) 558–571.
[14] R. Batuwita, V. Palade, Adjusted geometric-mean: a novel performance measure for imbalanced bioinformatics datasets learning, Journal of Bioinformatics and Computational Biology 10 (4) (2012).
[15] R. Batuwita, V. Palade, Class imbalance learning methods for support vector machines, in: H. He, Y. Ma (Eds.), Imbalanced Learning: Foundations, Algorithms, and Applications, Wiley, 2013, pp. 83–96.
[16] S. Bickel, M. Brückner, T. Scheffer, Discriminative learning under covariate shift, Journal of Machine Learning Research 10 (2009) 2137–2155.

[17] J. Błaszczyński, M. Deckert, J. Stefanowski, S. Wilk, Integrating selective pre-processing of imbalanced data with ivotes ensemble, in: M. Szczuka, M. Kryszkiewicz, S. Ramanna, R. Jensen, Q. Hu (Eds.), Rough Sets and Current Trends in Computing, LNSC, vol. 6086, Springer, Berlin/Heidelberg, 2010, pp. 148–157.
[18] J.P. Bradford, C. Kunz, R. Kohavi, C. Brunk, C.E. Brodley, Pruning decision trees with misclassification costs, in: Proceedings of the 10th European Conference on Machine Learning (ECML'98), 1998, pp. 131–136.
[19] A.P. Bradley, The use of the area under the roc curve in the evaluation of machine learning algorithms, Pattern Recognition 30 (7) (1997) 1145–1159.
[20] C.E. Brodley, M.A. Friedl, Identifying mislabeled training data, Journal of Artificial Intelligence Research 11 (1999) 131–167.
[21] C. Bunkhumpornpat, K. Sinapiromsaran, C. Lursinsap, Safe–level–SMOTE: Safe–level–synthetic minority over–sampling TEchnique for handling the class imbalanced problem. In: Proceedings of the 13th Pacific–Asia Conference on Advances in Knowledge Discovery and Data Mining PAKDD'09, 2009, pp. 475–482.
[22] C. Bunkhumpornpat, K. Sinapiromsaran, C. Lursinsap, DBSMOTE: density-based synthetic minority over-sampling technique, Applied Intelligence 36 (3) (2012) 664–684.
[23] J.Q. Candela, M. Sugiyama, A. Schwaighofer, N.D. Lawrence, Dataset Shift in Machine Learning, The MIT Press, 2009.
[24] D.R. Carvalho, A.A. Freitas, A hybrid decision tree/genetic algorithm method for data mining, Information Sciences 163 (1–3) (2004) 13–35.
[25] P.K. Chan, S.J. Stolfo, Toward scalable learning with non-uniform class and cost distributions: a case study in credit card fraud detection, in: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98). 1998. pp. 164–168.
[26] E.Y. Chang, B. Li, G. Wu, K. Goh, Statistical learning for effective visual information retrieval, in: Proceedings of the 2003 International Conference on Image Processing (ICIP'03), vol. 3, 2003, pp. 609–612.
[27] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, Journal of Artificial Intelligent Research 16 (2002) 321–357.
[28] N.V. Chawla, D.A. Cieslak, L.O. Hall, A. Joshi, Automatically countering imbalance and its empirical relationship to cost, Data Mining and Knowledge Discovery 17 (2) (2008) 225–252.
[29] N.V. Chawla, N. Japkowicz, A. Kotcz, Editorial: special issue on learning from imbalanced data sets, SIGKDD Explorations 6 (1) (2004) 1–6.
[30] N.V. Chawla, A. Lazarevic, L.O. Hall, K.W. Bowyer, SMOTEBoost: Improving prediction of the minority class in boosting, in: Proceedings of 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'03), 2003, pp. 107–119.
[31] X. Chen, T. Fang, H. Huo, D. Li, Graph-based feature selection for object-oriented classification in VHR airborne imagery, IEEE Transactions on Geoscience and Remote Sensing 49 (1) (2011) 353–365.
[32] D.A. Cieslak, N.V. Chawla, Analyzing pets on imbalanced datasets when training and testing class distributions differ, in: Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD08). Osaka, Japan, 2008, pp. 519–526.
[33] D.A. Cieslak, N.V. Chawla, A framework for monitoring classifiers' performance: when and why failure occurs?, Knowledge and Information Systems 18 (1) (2009) 83–108
[34] G. Cohen, M. Hilario, H. Sax, S. Hugonnet, A. Geissbuhler, Learning from imbalanced data in surveillance of nosocomial infection, Artificial Intelligence in Medicine 37 (2006) 7–18.
[35] C. Cortes, V. Vapnik, Support vector networks, Machine Learning 20 (1995) 273–297.
[36] J. Davis, M. Goadrich, The relationship between precisionrecall and ROC curves, in: Proceedings of the 23th International Conference on Machine Learning (ICML'06), ACM, 2006, pp. 233–240.
[37] M. Denil, T. Trappenberg, Overlap versus imbalance, in: Proceedings of the 23rd Canadian Conference on Advances in Artificial Intelligence (CCAI'10), Lecture Notes on Artificial Intelligence, vol. 6085, 2010, pp. 220–231.
[38] P. Domingos, Metacost: a general method for making classifiers cost–sensitive, in: Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD'99), 1999, pp. 155–164.
[39] D.J. Drown, T.M. Khoshgoftaar, N. Seliya, Evolutionary sampling and software quality modeling of high-assurance systems, IEEE Transactions on Systems, Man, and Cybernetics, Part A 39 (5) (2009) 1097–1107.
[40] C. Drummond, R.C. Holte, Cost curves: an improved method for visualizing classifier performance, Machine Learning 65 (1) (2006) 95–130.
[41] P. Ducange, B. Lazzerini, F. Marcelloni, Multi-objective genetic fuzzy classifiers for imbalanced and cost-sensitive datasets, Soft Computing 14 (7) (2010) 713–728.
[42] C. Elkan, The foundations of cost–sensitive learning, in: Proceedings of the 17th IEEE International Joint Conference on Artificial Intelligence (IJCAI'01), 2001, pp. 973–978.
[43] A. Estabrooks, T. Jo, N. Japkowicz, A multiple resampling method for learning from imbalanced data sets, Computational Intelligence 20 (1) (2004) 18–36.
[44] W. Fan, S.J. Stolfo, J. Zhang, P.K. Chan, Adacost: misclassification cost-sensitive boosting, in: Proceedings of the 16th International Conference on Machine Learning (ICML'96), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999, pp. 97–105.
[45] A. Fernández, M.J. del Jesus, F. Herrera, On the 2-tuples based genetic tuning performance for fuzzy rule based classification systems in imbalanced data-sets, Information Sciences 180 (8) (2010) 1268–1291.
[46] A. Fernández, S. García, M.J. del Jesus, F. Herrera, A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets, Fuzzy Sets and Systems 159 (18) (2008) 2378–2398.
[47] A. Fernandez, S. García, J. Luengo, E. Bernadó-Mansilla, F. Herrera, Genetics-based machine learning for rule induction: state of the art, taxonomy and comparative study, IEEE Transactions on Evolutionary Computation 14 (6) (2010) 913–941.
[48] A. Fernández, V. López, M. Galar, M.J. del Jesus, F. Herrera, Analysing the classification of imbalanced data-sets with multiple classes: binarization techniques and ad-hoc approaches, Knowledge-Based Systems 42 (2013) 97–110.
[49] J.H. Friedman, R. Kohavi, Y. Yun, Lazy decision trees, in: Proceedings of the AAAI/IAAI, vol. 1, 1996, pp. 717–724.
[50] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for class imbalance problem: bagging, boosting and hybrid based approaches, IEEE Transactions on Systems, Man, and Cybernetics – part C: Applications and Reviews 42 (4) (2012) 463–484.
[51] S. García, J. Derrac, I. Triguero, C.J. Carmona, F. Herrera, Evolutionary-based selection of generalized instances for imbalanced classification, Knowledge Based Systems 25 (1) (2012) 3–12.
[52] S. García, A. Fernández, F. Herrera, Enhancing the effectiveness and interpretability of decision tree and rule induction classifiers with evolutionary training set selection over imbalanced problems, Applied Soft Computing 9 (2009) 1304–1314.
[53] S. García, F. Herrera, Evolutionary under-sampling for classification with imbalanced data sets: proposals and taxonomy, Evolutionary Computation 17 (3) (2009) 275–306.
[54] V. García, R.A. Mollineda, J.S. Sánchez, A new performance evaluation method for two-class imbalanced problems, in: Proceedings of the Structural and Syntactic Pattern Recognition (SSPR'08) and Statistical Techniques in Pattern Recognition (SPR'08), Lecture Notes on Computer Science, vol. 5342, 2008, pp. 917–925.
[55] V. García, R.A. Mollineda, J.S. Sánchez, On the k-NN performance in a challenging scenario of imbalance and overlapping, Pattern Analysis Applications 11 (3–4) (2008) 269–280.
[56] V. García, R.A. Mollineda, J.S. Sánchez, Theoretical analysis of a performance measure for imbalanced data, in: 20th International Conference on Pattern Recognition (ICPR'10), 2010, pp. 617–620.
[57] V. García, R.A. Mollineda, J.S. Sánchez, Classifier performance assessment in two-class imbalanced problems, Internal Communication. (2012).
[58] V. García, J.S. Sánchez, R.A. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Knowledge Based Systems 25 (1) (2012) 13–21.

[59] N. García-Pedrajas, J. Pérez-Rodríguez, M. García-Pedrajas, D. Ortiz-Boyer, C. Fyfe, Class imbalance methods for translation initiation site recognition in DNA sequences, Knowledge Based Systems 25 (1) (2012) 22–34.
[60] A. Globerson, C.H. Teo, A. Smola, S. Roweis, An adversarial view of covariate shift and a minimax approach, in: J. Quiñonero Candela, M. Sugiyama, A. Schwaighofer, N.D. Lawrence (Eds.), Dataset Shift in Machine Learning, The MIT Press, 2009, pp. 179–198.
[61] H. Guo, H.L. Viktor, Learning from imbalanced data sets with boosting and data generation: the DataBoost-IM approach, SIGKDD Explorations Newsletter 6 (2004) 30–39.
[62] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The WEKA data mining software: an update, SIGKDD Explorations 11 (1) (2009) 10–18.
[63] H. Han, W.Y. Wang, B.H. Mao, Borderline–SMOTE: a new over–sampling method in imbalanced data sets learning, in: Proceedings of the 2005 International Conference on Intelligent Computing (ICIC'05), Lecture Notes in Computer Science, vol. 3644, 2005, pp. 878–887.
[64] P.E. Hart, The condensed nearest neighbor rule, IEEE Transactions on Information Theory 14 (1968) 515–516.
[65] H. He, Y. Bai, E.A. Garcia, S. Li, ADASYN: adaptive synthetic sampling approach for imbalanced learning, in: Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IJCNN'08), 2008, pp. 1322–1328.
[66] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Transactions on Knowledge and Data Engineering 21 (9) (2009) 1263–1284.
[67] S. Hido, H. Kashima, Y. Takahashi, Roughly balanced bagging for imbalanced data, Statistical Analysis and Data Mining 2 (2009) 412–426.
[68] R.C. Holte, L. Acker, B.W. Porter, Concept learning and the problem of small disjuncts, in: Proceedings of the International Joint Conferences on Artificial Intelligence, IJCAI'89, 1989, pp. 813–818.
[69] S. Hu, Y. Liang, L. Ma, Y. He, MSMOTE: improving classification performance when training data is imbalanced, in: Proceedings of the 2nd International Workshop on Computer Science and Engineering (WCSE'09), vol. 2, 2009, pp. 13–17.
[70] J. Huang, C.X. Ling, Using AUC and accuracy in evaluating learning algorithms, IEEE Transactions on Knowledge and Data Engineering 17 (3) (2005) 299–310.
[71] N. Japkowicz, Concept-learning in the presence of between-class and within-class imbalances, in: E. Stroulia, S. Matwin (Eds.), Proceedings of the 14th Canadian Conference on Advances in Artificial Intelligence (CCAI'08), Lecture Notes in Computer Science, vol. 2056, Springer, 2001, pp. 67–77.
[72] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intelligent Data Analysis Journal 6 (5) (2002) 429–450.
[73] T. Jo, N. Japkowicz, Class imbalances versus small disjuncts, ACM SIGKDD Explorations Newsletter 6 (1) (2004) 40–49.
[74] M.V. Joshi, V. Kumar, R.C. Agarwal, Evaluating boosting algorithms to classify rare classes: comparison and improvements, in: Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM'01), IEEE Computer Society, Washington, DC, USA, 2001, pp. 257–264.
[75] T.M. Khoshgoftaar, J. Van Hulse, A. Napolitano, Comparing boosting and bagging techniques with noisy and imbalanced data, IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 41 (3) (2011) 552–568.
[76] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: one-sided selection, in: Proceedings of the 14th International Conference on Machine Learning (ICML'97), 1997, pp. 179–186.
[77] L.I. Kuncheva, J.J. Rodrguez, A weighted voting framework for classifiers ensembles, Knowledge and Information Systems (2013), http://dx.doi.org/10.1007/s10115-012-0586-6. in press.
[78] N. Kwak, Feature extraction for classification problems and its application to face recognition, Pattern Recognition 41 (5) (2008) 1718–1734.
[79] J. Laurikkala, Improving identification of difficult small classes by balancing class distribution, in: Proceedings of the 8th Conference on AI in Medicine in Europe: Artificial Intelligence Medicine (AIME'01), 2001, pp. 63–66.
[80] C. Li, Classifying imbalanced data using a bagging ensemble variation (BEV), Proceedings of the 45th Annual Southeast Regional Conference, vol. 45, ACM-SE ACM, New York, NY, USA, 2007, pp. 203–208.
[81] M. Lin, K. Tang, X. Yao, Dynamic sampling approach to training neural networks for multiclass imbalance classification, IEEE Transactions on Neural Networks and Learning Systems 24 (4) (2013) 647–660.
[82] W. Lin, J.J. Chen, Class-imbalanced classifiers for high-dimensional data, Briefings in Bioinformatics 14 (1) (2013) 13–26.
[83] C.X. Ling, C. Li, Data mining for direct marketing: Problems and solutions, in: Proceedings of the 4th International Conference on Knownledge Discovery and Data Mining (KDD'98), 1998, pp. 73–79.
[84] C.X. Ling, Q. Yang, J. Wang, S. Zhang, Decision trees with minimal costs, in: C.E. Brodley (Ed.), Proceedings of the 21st International Conference on Machine Learning (ICML'04), ACM International Conference Proceeding Series, vol. 69, ACM, 2004, pp. 69–77.
[85] X.-Y. Liu, J. Wu, Z.-H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Transactions on System, Man and Cybernetics B 39 (2) (2009) 539–550.
[86] H.-Y. Lo, C.-M. Chang, T.-H. Chiang, C.-Y. Hsiao, A. Huang, T.-T. Kuo, W.-C. Lai, M.-H. Yang, J.-J. Yeh, C.-C. Yen, S.-D. Lin, Learning to improve area-under-FROC for imbalanced medical data classification using an ensemble method, SIGKDD Explorations 10 (2) (2008) 43–46.
[87] V. López, A. Fernández, M.J. del Jesus, F. Herrera, A hierarchical genetic fuzzy system based on genetic programming for addressing classification with highly imbalanced and borderline data-sets, Knowledge-Based Systems 38 (2013) 85–104.
[88] V. López, A. Fernández, J.G. Moreno-Torres, F. Herrera, Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics, Expert Systems with Applications 39 (7) (2012) 6585–6608.
[89] J. Luengo, A. Fernández, S. García, F. Herrera, Addressing data complexity for imbalanced data sets: analysis of SMOTE-based oversampling and evolutionary undersampling, Soft Computing 15 (10) (2011) 1909–1936.
[90] R. Martín-Félez, R.A., Mollineda, On the suitability of combining feature selection and resampling to manage data complexity, in: Proceedings of the Conferencia de la Asociacin Espaola de Inteligencia Artificial (CAEPIA'09), Lecture Notes on Artificial Intelligence, vol. 5988, 2010, pp. 141–150.
[91] M.A. Mazurowski, P.A. Habas, J.M. Zurada, J.Y. Lo, J.A. Baker, G.D. Tourassi, Training neural network classifiers for medical decision making: the effects of imbalanced datasets on classification performance, Neural Networks 21 (2–3) (2008).
[92] G.J. McLachlan, Discriminant Analysis and Statistical Pattern Recognition, John Wiley and Sons, 2004.
[93] L. Mena, J.A. González, Symbolic one-class learning from imbalanced datasets: application in medical diagnosis, International Journal on Artificial Intelligence Tools 18 (2) (2009) 273–309.
[94] J.G. Moreno-Torres, F. Herrera, A preliminary study on overlapping and data fracture in imbalanced domains by means of genetic programming-based feature extraction, in: Proceedings of the 10th International Conference on Intelligent Systems Design and Applications (ISDA'10), 2010, pp. 501–506.
[95] J.G. Moreno-Torres, X. Llorà, D.E. Goldberg, R. Bhargava, Repairing fractures between data using genetic programming-based feature extraction: a case study in cancer diagnosis, Information Sciences 222 (2013) 805–823.
[96] J.G. Moreno-Torres, T. Raeder, R. Aláiz-Rodríguez, N.V. Chawla, F. Herrera, A unifying view on dataset shift in classification, Pattern Recognition 45 (1) (2012) 521–530.
[97] K. Napierala, J. Stefanowski, S. Wilk, Learning from imbalanced data in presence of noisy and borderline examples, in: Proceedings of the 7th International Conference on Rough Sets and Current Trends in Computing (RSCTC'10), Lecture Notes on Artificial Intelligence, vol. 6086, 2010, pp. 158–167.
[98] A. Orriols-Puig, E. Bernadó-Mansilla, Evolutionary rule-based systems for imbalanced datasets, Soft Computing 13 (3) (2009) 213–225.
[99] A. Orriols-Puig, E. Bernadó-Mansilla, D.E. Goldberg, K. Sastry, P.L. Lanzi, Facetwise analysis of XCS for problems with class imbalances, IEEE Transactions on Evolutionary Computation 13 (2009) 260–283.
[100] J. Platt, Fast training of support vector machines using sequential minimal optimization, in: Advances in Kernel Methods – Support Vector Learning, MIT Press, Cambridge, MA, 1998, pp. 42–65.
[101] R. Polikar, Ensemble based systems in decision making, IEEE Circuits and Systems Magazine 6 (3) (2006) 21–45.
[102] R.C. Prati, G.E.A.P.A., Batista, Class imbalances versus class overlapping: an analysis of a learning system behavior, in: Proceedings of the 2004 Mexican International Conference on Artificial Intelligence (MICAI'04), 2004, pp. 312–321.

[103] R.C. Prati, G.E.A.P.A. Batista, M.C. Monard, A survey on graphical methods for classification predictive performance evaluation, IEEE Transactions on Knowledge and Data Engineering 23 (11) (2011) 1601–1618.
[104] J.R. Quinlan, C4.5: Programs for Machine Learning., Morgan Kauffman, 1993.
[105] T. Raeder, G. Forman, N.V. Chawla, Learning from imbalanced data: evaluation matters, in: D.E. Holmes, L.C. Jain (Eds.), Data Mining: Found. and Intell. Paradigms, vol. ISRL 23, Springer-Verlag, 2012, pp. 315–331.
[106] S.J. Raudys, A.K. Jain, Small sample size effects in statistical pattern recognition: recommendations for practitioners, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (3) (1991) 252–264.
[107] P. Riddle, R. Segal, O. Etzioni, Representation design and brute-force induction in a boeing manufacturing domain, Applied Artificial Intelligence 8 (1994) 125–147.
[108] L. Rokach, Ensemble-based classifiers, Artificial Intelligence Review 33 (1) (2010) 1–39.
[109] J.A. Sáez, J. Luengo, F. Herrera, A first study on the noise impact in classes for fuzzy rule based classification systems, in: Proceedings of the 2010 IEEE International Conference on Intelligent Systems and Knowledge Engineering (ISKE'10), IEEE Press, 2010, pp. 153–158.
[110] R.E. Schapire, A brief introduction to boosting, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'99), 1999, pp. 1401–1406.
[111] C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, A. Folleco, An empirical study of the classification performance of learners on imbalanced and noisy software quality data, Information Sciences (2013), http://dx.doi.org/10.1016/j.ins.2010.12.016. in press.
[112] C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, A. Napolitano, RUSBoost: a hybrid approach to alleviating class imbalance, IEEE Transactions on System, Man and Cybernetics A 40 (1) (2010) 185–197.
[113] J.P. Shaffer, Modified sequentially rejective multiple test procedures, Journal of the American Statistical Association 81 (395) (1986) 826–831.
[114] H. Shimodaira, Improving predictive inference under covariate shift by weighting the log-likelihood function, Journal of Statistical Planning and Inference 90 (2) (2000) 227–244.
[115] J. Stefanowski, S. Wilk, Improving rule based classifiers induced by MODLEM by selective pre-processing of imbalanced data, in: Proceedings of the RSKD Workshop at ECML/PKDD'07, 2007, pp. 54–65.
[116] J. Stefanowski, S. Wilk, Selective pre-processing of imbalanced data for improving classification performance, in: Proceedings of the 10th International Conference on Data Warehousing and Knowledge, Discovery (DaWaK08), 2008, pp. 283–292.
[117] Y. Sun, M.S. Kamel, A.K.C. Wong, Y. Wang, Cost-sensitive boosting for classification of imbalanced data, Pattern Recognition 40 (12) (2007) 3358–3378.
[118] Y. Sun, A.K.C. Wong, M.S. Kamel, Classification of imbalanced data: a review, International Journal of Pattern Recognition and Artificial Intelligence 23 (4) (2009) 687–719.
[119] Y. Tang, Y.-Q. Zhang, N.V. Chawla, S. Kresser, SVMs modeling for highly imbalanced classification, IEEE Transactions on Systems, Man and Cybernetics, Part B 9 (1) (2009) 281–288.
[120] D. Tao, X. Tang, X. Li, X. Wu, Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (7) (2006) 1088–1099.
[121] K.M. Ting, The problem of small disjuncts: its remedy in decision trees, in: Proceedings of the 10th Canadian Conference on Artificial Intelligence (CCAI'94), 1994, pp. 91–97.
[122] K.M. Ting, A comparative study of cost-sensitive boosting algorithms, in: Proceedings of the 17th International Conference on Machine Learning (ICML'00), Stanford, CA, USA, 2000, pp. 983–990.
[123] K.M. Ting, An instance-weighting method to induce cost-sensitive trees, IEEE Transactions on Knowledge and Data Engineering 14 (3) (2002) 659–665.
[124] I. Tomek, Two modifications of CNN, IEEE Transactions on Systems Man and Communications 6 (1976) 769–772.
[125] C.-H. Tsai, L.-C. Chang, H.-C. Chiang, Forecasting of ozone episode days by cost-sensitive neural network methods, Science of the Total Environment 407 (6) (2009) 2124–2135.
[126] P.D. Turney, Cost-sensitive classification: empirical evaluation of a hybrid genetic decision tree induction algorithm, Journal of Artificial Intelligence Research 2 (1995) 369–409.
[127] J. Van Hulse, T.M. Khoshgoftaar, A. Napolitano, An empirical comparison of repetitive undersampling techniques, in: Proceedings of the 2009 IEEE International Conference on Information Reuse, Integration (IRI'09), 2009, pp. 29–34.
[128] B.X. Wang, N. Japkowicz, Imbalanced data set learning with synthetic samples, in: Proceedings of the IRIS Machine Learning Workshop, 2004.
[129] J. Wang, J. You, Q. Li, Y. Xu, Extract minimum positive and maximum negative features for imbalanced binary classification, Pattern Recognition 45 (3) (2012) 1136–1145.
[130] S. Wang, X. Yao, Diversity analysis on imbalanced data sets by using ensemble models, in: Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Data Mining (CIDM'09), 2009, pp. 324–331.
[131] S. Wang, X. Yao, Relationships between diversity of classification ensembles and single-class performance measures, IEEE Transactions on Knowledge and Data Engineering 25 (1) (2013) 206–219.
[132] Z. Wang, V. Palade, Building interpretable fuzzy models for high dimensional data analysis in cancer diagnosis, BMC Genomics 12 ((S2):S5) (2011).
[133] M. Wasikowski, X.-W. Chen, Combating the small sample class imbalance problem using feature selection, IEEE Transactions on Knowledge and Data Engineering 22 (10) (2010) 1388–1400.
[134] G.M. Weiss, Timeweaver: a genetic algorithm for identifying pre-dictive patterns in sequences of events, in: W. Banzhaf, J. Daida, A.E. Eiben, M.H. Garzon, V. Honavar, M. Jakiela, R.E. Smith (Eds.), Proceedings of the Genetic and Evolutionary Computation Conference GECCO'99, vol. 1, Morgan Kaufmann, Orlando, Florida, USA, 1999, pp. 718–725.
[135] G.M. Weiss, Mining with rarity: a unifying framework, SIGKDD Explorations 6 (1) (2004) 7–19.
[136] G.M. Weiss, Mining with rare cases, in: O. Maimon, L. Rokach (Eds.), The Data Mining and Knowledge Discovery Handbook, Springer, 2005, pp. 765–776.
[137] G.M. Weiss, The impact of small disjuncts on classifier learning, in: R. Stahlbock, S.F. Crone, S. Lessmann (Eds.), Data Mining: Annals of Information Systems, vol. 8, Springer, 2010, pp. 193–226.
[138] G.M. Weiss, F.J. Provost, Learning when training data are costly: the effect of class distribution on tree induction, Journal of Artificial Intelligence Research 19 (2003) 315–354.
[139] G.M. Weiss, Y. Tian, Maximizing classifier utility when there are data acquisition and modeling costs, Data Mining and Knowledge Discovery 17 (2) (2008) 253–282.
[140] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man and Cybernetics 2 (3) (1972) 408–421.
[141] R. Yan, Y. Liu, R. Jin, A. Hauptmann, On predicting rare classes with SVM ensembles in scene classification, in: Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'03), vol. 3, 2003, pp. 21–24.
[142] P. Yang, L. Xu, B.B. Zhou, Z. Zhang, A.Y. Zomaya, A particle swarm based hybrid system for imbalanced medical data sampling, BMC Genomics 10 (Suppl. 3) (2009). art. no. S34..
[143] Q. Yang, X. Wu, 10 challenging problems in data mining research, International Journal of Information Technology and Decision Making 5 (4) (2006) 597–604.
[144] Y. Yang, X. Wu, X. Zhu, Conceptual equivalence for contrast mining in classification learning, Data & Knowledge Engineering 67 (3) (2008) 413–429.
[145] S. Yen, Y. Lee, Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset, in: Proceedings of the 2006 International Conference on Intelligent, Computing (ICIC06), 2006, pp. 731–740.

[146] K. Yoon, S. Kwek, An unsupervised learning approach to resolving the data imbalanced issue in supervised learning problems in functional genomics, in: Proceedings of the 5th International Conference on Hybrid Intelligent Systems (HIS'05), 2005, pp. 303–308.
[147] B. Zadrozny, C. Elkan, Learning and making decisions when costs and probabilities are both unknown, in: Proceedings of the 7th International Conference on Knowledge Discovery and Data Mining (KDD'01), 2001, pp. 204–213.
[148] B. Zadrozny, J. Langford, N. Abe, Cost–sensitive learning by cost–proportionate example weighting, in: Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03), 2003, pp. 435–442.
[149] J. Zhang, I. Mani, KNN approach to unbalanced data distributions: a case study involving information extraction, in: Proceedings of the 20th International Conference on Machine Learning (ICML'03), Workshop Learning from Imbalanced Data Sets, 2003.
[150] Z.-H. Zhou, X.-Y. Liu, Training cost-sensitive neural networks with methods addressing the class imbalance problem, IEEE Transactions on Knowledge and Data Engineering 18 (1) (2006) 63–77.
[151] X. Zhu, X. Wu, Class noise vs. attribute noise: a quantitative study, Artificial Intelligence Review 22 (3) (2004) 177–210.
[152] W. Zong, G.-B. Huang, Y. Chen, Weighted extreme learning machine for imbalance learning, Neurocomputing 101 (2013) 229–242.

2. Addressing the Data Intrinsic Characteristics of Imbalanced Problems using FRBCSs and Machine Learning Techniques

The journal papers associated to this part are:

2.1. A Hierarchical Genetic Fuzzy System Based On Genetic Programming for Addressing Classification with Highly Imbalanced and Borderline Data-sets

- V. López, A. Fernández, M. J. del Jesus, F. Herrera, A Hierarchical Genetic Fuzzy System Based On Genetic Programming for Addressing Classification with Highly Imbalanced and Borderline Data-sets. Knowledge-Based Systems 38 (2013) 85–104, doi: 10.1016/j.knosys.2012.08.025

  - Status: **Published**.
  - Impact Factor (JCR 2012): 4.104.
  - Subject Category: Computer Science, Artificial Intelligence. Ranking 6 / 115 (**Q1**).

Knowledge-Based Systems 38 (2013) 85–104

Contents lists available at SciVerse ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A hierarchical genetic fuzzy system based on genetic programming for addressing classification with highly imbalanced and borderline data-sets

Victoria López [a,*], Alberto Fernández [b], María José del Jesus [b], Francisco Herrera [a]

[a] Dept. of Computer Science and Artificial Intelligence, CITIC-UGR, Research Center on Information and Communications Technology, University of Granada, 18071 Granada, Spain
[b] Dept. of Computer Science, University of Jaén, 23071 Jaén, Spain

ARTICLE INFO

Article history:
Received 18 October 2011
Received in revised form 3 June 2012
Accepted 24 August 2012
Available online 3 October 2012

Keywords:
Fuzzy rule based classification systems
Hierarchical fuzzy partitions
Genetic rule selection
Tuning
Imbalanced data-sets
Borderline examples

ABSTRACT

Lots of real world applications appear to be a matter of classification with imbalanced data-sets. This problem arises when the number of instances from one class is quite different to the number of instances from the other class. Traditionally, classification algorithms are unable to correctly deal with this issue as they are biased towards the majority class. Therefore, algorithms tend to misclassify the minority class which usually is the most interesting one for the application that is being sorted out.

Among the available learning approaches, fuzzy rule-based classification systems have obtained a good behavior in the scenario of imbalanced data-sets. In this work, we focus on some modifications to further improve the performance of these systems considering the usage of information granulation. Specifically, a positive synergy between data sampling methods and algorithmic modifications is proposed, creating a genetic programming approach that uses linguistic variables in a hierarchical way. These linguistic variables are adapted to the context of the problem with a genetic process that combines rule selection with the adjustment of the lateral position of the labels based on the 2-tuples linguistic model.

An experimental study is carried out over highly imbalanced and borderline imbalanced data-sets which is completed by a statistical comparative analysis. The results obtained show that the proposed model outperforms several fuzzy rule based classification systems, including a hierarchical approach and presents a better behavior than the C4.5 decision tree.

1. Introduction

Learning from imbalanced data-sets is an issue that has attracted a lot of attention in machine learning research [29,51]. This problem is characterized by a class distribution where the number of examples in one class is outnumbered by the number of examples in the other class. The presence of imbalanced data-sets is dominant in a high number of real problems including, but not limited to, medical diagnosis, fraud detection, finances, risk management, network intrusion and so on. Additionally, the positive or minority class is usually the one that has the highest interest from the learning point of view and it also implies a great cost when it is not well classified [17,57].

A standard classifier that seeks accuracy over a full range of instances is frequently not suitable to deal with imbalanced learning tasks, since it tends to be overwhelmed by the majority class thus misclassifying the minority examples. This situation becomes critical when the minority class is greatly outnumbered by the majority class, generating an scenario of highly imbalanced data-sets where the performance deterioration is amplified. However, some studies have shown that imbalance for itself is not the only factor that hinders the classification performance [37]. There are several data intrinsic characteristics which lower the learning effectiveness. Some of these handicaps within the data are the presence of small disjuncts [53], the overlap between the classes [26] or the existence of noisy [49] and borderline [44] samples. There is no need to say that when the classification data share an skewed data distribution together with any of the aforementioned situations, the performance degradation is intensified [19,42,53].

A large number of approaches have been proposed to deal with the class imbalance problem. Those solutions fall largely into two major categories. The first is data sampling in which the training data distribution is modified to obtain a set with a balanced distribution. Standard classifiers are thus helped to obtain a correct identification of data [9,6]. The second is through algorithmic modification where the base learning methods are modified to consider the imbalanced distribution of the data. In this manner, base learning methods change some of its internal operations accordingly [57].

Fuzzy Rule-Based Classification Systems (FRBCSs) [34] are useful and well-known tools in the machine learning framework. They provide a good trade-off between the empirical precision of

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: vlopez@decsai.ugr.es (V. López), alberto.fernandez@ujaen.es (A. Fernández), mjjesus@ujaen.es (M.J. del Jesus), herrera@decsai.ugr.es (F. Herrera).

0950-7051/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2012.08.025

traditional engineering techniques and the interpretability achieved through the use of linguistic labels whose semantic is close to the natural language. Specifically, recent works have shown that FRBCSs have a good behavior dealing with imbalanced data-sets by means of the application of instance preprocessing techniques [20].

The hybridization between fuzzy logic and genetic algorithms leading to Genetic Fuzzy Systems (GFSs) [12,30] is one of the most popular approaches used when different computational intelligence techniques are combined. A GFS is basically a fuzzy system augmented by a learning process based on evolutionary computation. Among evolutionary algorithms, Genetic Programming (GP) [39] is a development of classical genetic algorithms that evolve tree-shaped solutions using variable length chromosomes. GP has been used in FRBCSs to learn fuzzy rule bases [7] profitting from its high expressive power and flexibility.

However, the disadvantage of FRBCSs is the inflexibility of the concept of linguistic variable because it imposes hard restrictions on the fuzzy rule structure [5] which may suppose a loss in accuracy when dealing with some complex systems, such as high dimensional problems, the presence of noise or overlapped classes. Many different possibilities to enhance the linguistic fuzzy modeling have been considered in the specialized literature. All of these approaches share the common idea of improving the way in which the linguistic fuzzy model performs the interpolative reasoning by inducing a better cooperation among the rules in the Knowledge Base (KB). This rule cooperation may be induced acting on three different model components:

- Approaches acting on the whole KB. This includes the KB derivation [43] and a hierarchical linguistic rule learning [14].
- Approaches acting on the Rule Base (RB). The most common approach is rule selection [35] but also multiple rule consequent learning [11] could be considered.
- Approaches acting on the Data Base (DB). For example a priori granularity learning [13] or membership function tuning [1].

In this work, we present a procedure to obtain an Hierarchical Fuzzy Rule Based Classification System (HFRBCS) to deal with imbalanced data-sets. In order to do so, this model introduces modifications both at the data and algorithm level. This procedure is divided into three different steps:

1. A preprocessing technique, the Synthetic Minority Oversampling Technique (SMOTE) [9], is used to balance the distribution of training examples in both classes.
2. A hierarchical knowledge base (HKB) [14] is generated, using the GP-COACH (Genetic Programming-based learning of COmpact and ACcurate fuzzy rule-based classification systems for High-dimensional problems) algorithm [7] to build the RB. The GP-COACH algorithm has been modified to extend a classical KB into a HKB, integrating a rule expansion process to create high granularity rules in each generation of the algorithm. The usage of a HKB implies an adaptation of the components to allow the interaction of the different granularities in the RB population.
3. A post-processing step involving rule selection and the application of the 2-tuples based genetic tuning is applied to improve the overall performance.

The combination of these steps constitutes a convenient approach to solve the problem of classification with imbalanced data-sets. First of all, the preprocessing technique compensates the number of instances for each class easing the learning process for the consequent procedures. Then, the step to learn the HKB is used to address the imbalanced problem together with some of the data intrinsic characteristics that difficult the learning. This HKB process is appropriate because it increases the accuracy by reinforcing those problem subspaces that are specially difficult in this environment, such as borderline instances [44], small disjuncts [37] or overlapping regions [26]. Finally, the post-processing step refines the results achieved by the previous process. The integration of these schemes completes our proposal, which will be denoted as GP-COACH-H (GP-COACH Hierarchical).

We will focus on two difficult situations in the scenario of imbalanced data, such as highly imbalanced and borderline imbalanced classification problems. For that, we have selected a benchmark of 44 and 30 problems respectively from KEEL data-set repository[1] [2]. We will perform our experimental analysis focusing on the precision of the models using the Geometric Mean of the true rates (GM) [4]. This study will be carried out using non-parametric tests to check whether there are significant differences among the obtained results [25].

This work is structured in the following way. First, Section 2 presents an introduction of classification with imbalanced problems, describing its features, the SMOTE algorithm and the metrics that are used in this framework. Next, Section 3 introduces the proposed approach. Sections 4 and 5 describe the experimental framework used and the analysis of results, respectively. Next, the conclusions achieved in this work are shown in Section 6. Finally, we include an appendix with the detailed results for the experiments performed in the experimental study.

2. Imbalanced data-sets in classification

In this section we delimit the context in which this work is content, briefly introducing the problem of imbalanced classification. Then, we will describe the preprocessing technique that we have applied in order to deal with the imbalanced data-sets: the SMOTE algorithm [9]. We finish this section describing the evaluation metrics that are used in this specific problem with respect to the most common ones in classification.

2.1. The problem of imbalanced data-sets

In some classification problems, the number of examples that represent the diverse classes is very different. Specifically, the imbalance problem occurs when one class is represented only by a few number of examples, while the others are represented by a large number of examples [51,29]. In this paper, we focus on two-class imbalanced data-sets, where there is a positive (minority) class, with the lowest number of instances, and a negative (majority) class, with the highest number of instances.

This problem is prevalent in many real world applications, such as medical diagnosis [45,48], anomaly detection [38], image analysis [8] or bioinformatics [28], just referencing some of them. Furthermore, it is usual that positive classes are the most interesting from the application point of view so it is crucial to correctly identify these cases. The importance of this problem in the aforementioned uses has increased the attention towards it, which has been considered one of the 10 challenging problems in data mining [56].

Although these issues occur frequently in data, many data mining methods do not naturally perform well under these circumstances. In fact, many only work optimally when the classes in data are relatively balanced. Furthermore, the performance of algorithms is usually more degraded when the imbalance increases because positive examples are more easily forgotten. That situation is critical in highly imbalanced data-sets because the number of

[1] http://www.keel.es/datasets.php.

positive instances in the data-set is negligible and that situation increases the difficulty that most learning algorithms have in detecting positive regions. Figs. 1 and 2 depict two data-sets with low imbalance and high imbalance respectively.

However, the imbalanced data-set is also affected by some circumstances that make the learning more difficult. For example, metrics that have been used traditionally seem inappropriate in this scenario when they ascribe a high performance to a trivial classifier that predicts all samples as negative. This behavior is wrapped up in the inner way of building an accurate model, preferring general rules with good coverage for the negative class and disregarding more specific rules which are the ones associated to the positive class.

An important issue that appear in imbalanced data-sets is the presence of borderline examples. Inspired by Kubat and Matwin [40] we may distinguish between safe, noisy and borderline examples. Safe examples are placed in relatively homogeneous areas with respect to the class label. By noisy examples we understand individuals from one class occurring in safe areas of the other class. Finally, borderline examples are located in the area surrounding class boundaries, where the positive and negative classes overlap. These borderline examples make difficult to determine a correct discrimination of the classes. For instance, Napierala et al. [44] present in a series of experiments in which it is shown that the degradation in performance of a classifier in an imbalanced scenario is strongly affected by the number of borderline examples.

2.2. Addressing imbalanced data-sets: use of preprocessing and SMOTE algorithm

A large number of approaches have been proposed to deal with the class-imbalance problem [51,41,42]. These approaches can be categorized in two groups: the internal approaches that create new algorithms or modify existing ones to take the class-imbalance problem into consideration [4] and external approaches that preprocess the data in order to diminish the effect of their class imbalance [6,23,27]. Furthermore, cost-sensitive learning solutions incorporating both approaches assume higher misclassification costs with samples in the positive class and seek to minimize the high cost errors [17,57]. The great advantage of the external approaches is that they are more versatile, since their use is independent of the classifier selected. Furthermore, we may preprocess all data-sets before-hand in order to use them to train different classifiers. In this manner, the computation time needed to prepare the data is only required once. According to this, in this work we have chosen an oversampling method which is a reference in this area: the SMOTE algorithm [9] and a variant called SMOTE + ENN [6].

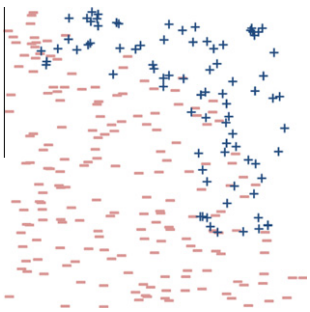


Fig. 1. Data-set with low imbalance (IR = 2.23).

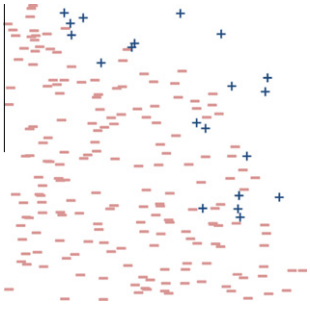


Fig. 2. Data-set with high imbalance (IR = 9.15).

In this approach, the positive class is over-sampled by taking each positive class sample and introducing synthetic examples along the line segments joining any/all of the k positive class nearest neighbors. Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen. This process is illustrated in Fig. 3, where x_i is the selected point, x_{i1} to x_{i4} are some selected nearest neighbors and r_1 to r_4 the synthetic data points created by the randomized interpolation.

Synthetic samples are generated in the following way: take the difference between the feature vector (sample) under consideration and its nearest neighbor. Multiply this difference by a random number between 0 and 1, and add it to the feature vector under consideration. This causes the selection of a random point along the line segment between two specific features. This approach effectively forces the decision region of the positive class to become more general. An example is detailed in Fig. 4.

In short, its main feature is to form new positive class examples by interpolating between several positive class examples that lie together. Thus, the overfitting problem is avoided and causes the decision boundaries for the positive class to spread further into the negative class space.

Nevertheless, class clusters may be not well defined in cases where some negative class examples might be invading the positive class space. The opposite can also be true, since interpolating positive class examples can expand the positive class clusters, introducing artificial positive class examples too deeply into the negative class space. Inducing a classifier in such a situation can lead to over-fitting. For this reason we will also consider in this work a hybrid approach, "SMOTE+ENN", where the Wilson's Edited Nearest Neighbor Rule [54] is used after the SMOTE application to remove any example from the training set misclassified by its three nearest neighbors.

2.3. Evaluation in imbalanced domains

The measures of the quality of classification are built from a confusion matrix (shown in Table 1) which records correctly and incorrectly recognized examples for each class.

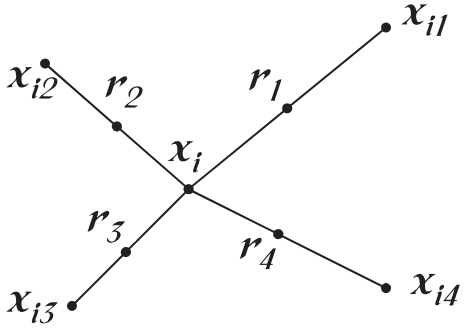


Fig. 3. An illustration of how to create the synthetic data points in the SMOTE algorithm.

```
Consider a sample (6,4) and let (4,3) be its nearest neighbor.
(6,4) is the sample for which k-nearest neighbors
are being identified and (4,3) is one of its k-nearest neighbors.
Let: f1_1 = 6 f2_1 = 4, f2_1 - f1_1 = -2
f1_2 = 4 f2_2 = 3, f2_2 - f1_2 = -1
The new samples will be generated as
(f1',f2') = (6,4) + rand(0-1) * (-2,-1)
rand(0-1) generates a random number between 0 and 1.
```

Fig. 4. Example of the SMOTE application.

The most used empirical measure, accuracy (Eq. (1)), does not distinguish between the number of correct labels of different classes, which in the ambit of imbalanced problems may lead to erroneous conclusions. For example a classifier that obtains an accuracy of 90% in a data-set with a 90% of negative instances, might not be accurate if it does not cover correctly any positive class instance.

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \tag{1}$$

Because of this, instead of using accuracy, more appropriate metrics in this situation are considered. Two common measures, sensitivity and specificity (Eqs. (2) and (3)), approximate the probability of the positive (negative) label being true. In other words, they assess the effectiveness of the algorithm on a single class.

$$sensitivity = \frac{TP}{TP + FN} \tag{2}$$

$$specificity = \frac{TN}{FP + TN} \tag{3}$$

The metric used in this work is the geometric mean of the true rates [4,40], which can be defined as

$$GM = \sqrt{\frac{TP}{TP + FN} \cdot \frac{TN}{FP + TN}} \tag{4}$$

This metric attempts to maximize the accuracy of each one of the two classes with a good balance. It is a performance metric that links both objectives.

3. The hierarchical genetic programming fuzzy rule based classification system with rule selection and tuning (GP-COACH-H)

In this section, we will describe our proposal to obtain a hierarchical FRBCS through the usage of GP and applying rule selection together with 2-tuples lateral tuning, denoted as GP-COACH-H. This proposal is defined through its components in the following way: Section 3.1 presents a brief introduction of FRBCSs in order to contextualize the algorithm; next, Section 3.2 describes the GP-COACH algorithm [7] which is the linguistic rule generation method based on GP that we have used as base for our proposal of a hierarchical rule base generation method; later, in Section 3.3, the building of the hierarchical fuzzy rule based classification is detailed, mentioning the modifications the hierarchical procedure introduces in the knowledge base generation and in the basic running of the GP-COACH algorithm; subsequently, Section 3.4 shows the selection of the best cooperative rules and the tuning of the databases in a genetic process where both objectives collaborate; and finally, Section 3.5 summarizes the description of the proposal.

Table 1
Confusion matrix for a two-class problem.

	Positive prediction	Negative prediction
Positive class	True Positive (TP)	False Negative (FN)
Negative class	False Positive (FP)	True Negative (TN)

3.1. Fuzzy rule based classification systems

FRBCSs are useful and well-known tools in the machine learning framework since they can provide an interpretable model for the end user. A FRBCS has two main components: the Inference System and the KB. In a linguistic FRBCS, the KB is composed of a RB, constituted by a set of fuzzy rules, and the DB that stores the membership functions of the fuzzy partitions associated to the input variables. If expert knowledge of the problem is not available, it is necessary to use some Machine Learning process to obtain the KB from examples.

Any classification problem consists of N training patterns $\mathbf{x}_p = (x_{p1}, \ldots, x_{pn})$, $p = 1,2,\ldots,m$ from M classes where x_{pi} is the ith attribute value ($i = 1,2,\ldots,n$) of the pth training pattern.

In this work, we use fuzzy rules of the following form to build our classifier:

$$\text{Rule } R_j: \text{If } x_1 \text{ is } \widehat{A}_1^j \text{ and } \ldots \text{ and } x_n \text{ is } \widehat{A}_n^j \text{ then Class} = C_j \text{ with } RW_j \tag{5}$$

where R_j is the label of the jth rule, $\mathbf{x} = (x_1, \ldots, x_n)$ is an n-dimensional pattern vector, $\widehat{A}_i^j$ is a set of linguistic labels $\{L_i^1 \text{ or } \ldots \text{ or } L_i^{l_k}\}$ joined by a disjunctive operator, C_j is a class label, and RW_j is the rule weight [33]. We use triangular membership functions as linguistic labels whose combination will form an antecedent fuzzy set. This kind of rule is called a DNF fuzzy rule.

To compute the rule weight, many heuristics have been proposed [36]. In our proposal, we compute the rule weight as the fuzzy confidence or Certainty Factor (CF) [15], showed in Eq. (6):

$$RW_j = CF_j = \frac{\sum_{x_p \in Class_j} \mu_{\widehat{A}_j}(x_p)}{\sum_{p=1}^{N} \mu_{\widehat{A}_j}(x_p)} \tag{6}$$

where $\mu_{\widehat{A}_j}(x_p)$ is the matching degree of the pattern x_p with the antecedent part of the fuzzy rule R_j.

GP-COACH-H uses the normalized sum fuzzy reasoning method [15] for classifying new patterns by the RB, a general reasoning model for combining information provided by different rules, where each rule promotes the classification with its consequent class according to the matching degree of the pattern with the antecedent part of the fuzzy rule together with its weight. The total sum for each class is computed as follows:

$$Sum_{Class\ h}(x_p) = \frac{\sum_{R_j \in RB, C_j = h} \mu_{\widehat{A}_j}(x_p) \cdot CF_j}{\max_{c=1,\ldots,M} \sum_{R_j \in RB, C_j = c} \mu_{\widehat{A}_j}(x_p) \cdot CF_j} \tag{7}$$

$$Class(x_p) = \arg\max(Sum_{Class\ h}(x_p)) \tag{8}$$

3.2. The GP-COACH algorithm

The GP-COACH algorithm [7] is a genetic programming-based algorithm for the learning of fuzzy rule bases. We will use this method as a base for our hierarchical model modifying its behavior to include the different granularity levels into its inner way of running.

This algorithm is a genetic cooperative-competitive learning approach where the whole population represents the RB obtained. Each individual in the population codifies a rule. These rules are DNF fuzzy rules (Eq. (5)) which allow the absence of some input features and are generated according to the production rules of a context-free grammar. As DB we are using linguistic partitions

with the same number of linguistic terms for all input variables, composed of symmetrical triangular-shaped and uniformly distributed membership functions.

There are two evaluation functions in the GP-COACH algorithm: a local fitness function, known as *raw_fitness*, to evaluate the performance of each rule and a global fitness function, known as *global_fitness*, to evaluate the behavior of the whole rule population. The *raw_fitness* is computed according to *Confidence* (shown in Eq. (6)) and *Support*, which measure the accuracy of the rule and the extent of knowledge of the rule respectively:

$$Support(R_j) = \frac{\sum_{x_p \in Class C_j} \mu_{A_j}(x_p)}{N_{C^j}} \quad (9)$$

where N_{C^j} is the number of examples that belong to the same class that the one determined in the consequent of the rule. Therefore, the *raw_fitness* is computed in the following way:

$$raw\_fitness(R_j) = \alpha \cdot Confidence(R_j) + (1 - \alpha) \cdot Support(R_j) \quad (10)$$

Finally, it is important to point out that each time that an individual is evaluated it is also necessary to modify its certainty degree. On the other hand, the *global_fitness* score measure is defined as follows:

$$global\_fitness = w_1 \cdot accuracy + w_2 \cdot (1.0 - Var_N) + w_3 \cdot (1.0 - Cond_N) + w_4 \cdot (1.0 - Rul_N) \quad (11)$$

where Var_N and $Cond_N$ are the normalized values of the average number of variables and conditions in the rules, and Rul_N is the normalized number of rules in the population respectively.

The GP-COACH algorithm also includes a mechanism for maintaining the diversity in the population: the token competition procedure [55], inspired by the following natural behavior: when an individual finds a good place to live, it will maintain its position there preventing the others to share its position unless they are stronger. Each example in the training set is called a token and the rules in the population compete to acquire as many tokens as possible. When a rule matches an example, it tries to seize the token, however, this token will be assigned to the stronger rule that matches the example. Stronger individuals exploit their dominant position by seizing as many tokens as they can. The other ones entering the same position will have their strength decreased because they cannot compete with the stronger ones, by the addition of a penalization in the fitness score of the individual. Therefore, to model this behavior, a *penalized_function* is defined:

$$penalized\_fitness(R_j) = \begin{cases} raw\_fitness(R_j) \cdot \frac{count(R_j)}{ideal(R_j)} & \text{if } ideal(R_j) > 0, \\ 0, & \text{otherwise} \end{cases} \quad (12)$$

where $raw\_fitness(R_j)$ is the fitness score obtained from the evaluation function (Eq. (10)), $count(R_j)$ is the number of tokens that the individual actually seized and $ideal(R_j)$ is the total number of tokens that it can seize, which is equal to the number of examples that the individual matches.

As a result of the token competition, there can be individuals that cannot grab any token. These individuals are considered as irrelevant, and they are eliminated from the population because all of their examples are covered by other stronger individuals.

Once the token competition mechanism has been applied, it is possible that some of the examples in the training set are not covered by any of the rules in the population. The generation of new specific rules covering these examples improves the diversity in the population, and helps the evolutionary process to easily find stronger and more general rules covering these examples. Therefore, GP-COACH learns rule sets having two different types of fuzzy rules: a core of strong and general rules (primary rules) that cover most of the examples, and a small set of weaker and more specific rules (secondary rules) that are only used if there are not any primary rule matching the example. These secondary rules are generated by the Chi et al. algorithm [10] over the set of training examples that are left uncovered by the primary rules. This scaly scheme is used in rule based algorithms to cover in a better way the data space [52]. GP-COACH uses four different genetic operators to generate new individuals during the evolutionary process:

1. *Crossover*: A part in the first parent is randomly selected and exchanged by another part, randomly selected, in the second one.
2. *Mutation*: It is applied to a variable in the rule randomly chosen. The mutation can add a new label to the label set associated to the variable, remove a label from the label set associated to the variable or exchange one label in the label set associated to the variable with another one not included.
3. *Insertion*: It adds a new variable to the parent rule with at least one linguistic label.
4. *Dropping condition*: It selects one variable and removes its conditions from the rule.

These operations only generate one offspring each time they are applied.

Fig. 5 shows the pseudocode associated to the GP-COACH algorithm. This method begins creating a random initial population according to the rules in the context-free grammar. Each individual in this population is then evaluated. After that, the initial population is kept as the best evolved population and its global fitness score is computed. Then, the initial population is copied to the current population and the evolutionary process begins:

1. An offspring population, with the same size than the current one, is created. Parents are selected by using the tournament selection mechanism and children are created by using one of the four genetic operators. The genetic operator selection is done in a probabilistic way according to a given probability.
2. Once the offspring population is created, it is joined to the current population, creating a new population whose size is double the current population size. Individuals in this new population are sorted according to their fitness and the token competition mechanism is applied. Secondary rules are created if some examples remain uncovered.

```
1  CurrentPop = InitPopulation();
2  EvaluatePopRules (CurrentPop);
3  CurrentFitness = EvaluatePopulation (CurrentPop);
4  BestPop = CurrentPop;
5  BestFitness = CurrentFitness;
6  N = 0;
7  while N < N_eval do
8      OffspringsPop = ∅;
9      while OffspringsPop.size() ≠ CurrentPop.size() do
10         ParentRule = tournamentSelection (CurrentPop);
11         GeneratedRule = geneticOperator (ParentRule, CurrentPop);
12         OffspringsPop.add(GeneratedRule);
13     end
14     EvaluatePopRules (OffspringsPop);
15     JointPop = CurrentPop ∪ OffspringsPop;
16     JointPop = tokenCompetition (JointPop);
17     CurrentPop = JointPop ∪ generateSecondaryRules (JointPop);
18     CurrentFitness = EvaluatePopulation (CurrentPop);
19     if CurrentFitness > BestFitness then
20         BestPop = CurrentPop;
21         BestFitness = CurrentFitness;
22     end
23     N++;
24 end
   Output: BestPop
```

Fig. 5. The GP-COACH algorithm.

3. The global fitness score measure is then calculated for this new population. We check whether this new fitness is better than the one stored for the best population, updating the best population and fitness if necessary. In any case, the new population is copied as the current population in order to be able to apply the evolutionary process again.

The evolutionary process ends when the stop condition is verified, that is when a number of evaluations is reached. Then, the population kept as the best one is returned as the solution to the problem and GP-COACH finishes.

3.3. Hierarchical fuzzy rule based classification system construction

HFRBCs try to improve the performance of fuzzy rule based systems in data subspaces that are particularly difficult. In order to do so, instead of the classical definition of the KB, we use an extension known as HKB [14], which is composed of a set of layers. We will divide this subsection in two parts: the first part is devoted to the presentation of the HKB, its components and some general guidelines about how to build it; the second part is devoted to the integration of the HKB into the inner way of running of the GP-COACH algorithm which we have used as base for our proposal.

3.3.1. Hierarchical knowledge base

In order to overcome the inflexibility of the concept of linguistic variable which degrades the performance of algorithms in complex search spaces, we extend the definition of the standard KB into an hierarchical one that preserves the original model descriptive power and increases its accuracy. This HKB is composed of a set of layers. We define a layer by its components in the following way:

$$layer(t, n(t)) = DB(t, n(t)) + RB(t, n(t)) \tag{13}$$

with $n(t)$ being the number of linguistic terms that compose the partitions of layer, $DB(t,n(t))$ (*t-linguistic partitions*) being the DB which contains the linguistic partitions with granularity level $n(t)$ of layer, and $RB(t,n(t))$ (*t-linguistic rules*) being the RB formed by those linguistic rules whose linguistic variables take values in the former partitions. The number of linguistic terms in the *t-linguistic partitions* is defined in the following way:

$$n(t) = (n(1) - 1) \cdot 2^{t-1} + 1 \tag{14}$$

with $n(1)$ being the granularity of the initial fuzzy partitions.

This set of layers is organized as a hierarchy, where the order is given by the granularity level of the linguistic partition defined in each layer. That is, given two successive layers t and $t+1$ then the granularity level of the linguistic partitions of layer $t+1$ is greater than the ones of layer t. This causes a refinement of the previous layer linguistic partitions. As a consequence of the previous definitions, we can now define the HKB as the union of every layer t

$$HKB = \bigcup_t layer(t, n(t)) \tag{15}$$

Our proposal considers a two-layer HKB, i.e. starting with an initial layer t, we produce layer $t+1$ in order to extract the final system of linguistic rules. This fact allows the approach to build a significantly more accurate modeling of the problem space.

First of all, we need to build the two-layer HDB. The first level layer is built by the usage of *linguistic partitions* with the same number of linguistic terms for all input variables, composed of symmetrical triangular-shaped and uniformly distributed membership functions. The second layer, is built preserving all the membership function modal points, corresponding to each linguistic term, through the higher layers of the hierarchy and adding a new linguistic term between each two consecutive terms of the *t-linguistic partition* reducing the support of these linguistic terms in order to keep place for the new one, which is located in the middle of them. Fig. 6 shows the *linguistic partitions* from one level to another, with $n(1) = 3$ and $n(2) = 5$.

The second step affects the generation of the HRB which is composed by the RB of layer t and a RB of layer $t+1$. Two measures of error are usually used to build a RB of layer $t+1$ from a layer RB of layer t: a global measure, which is used to evaluate the complete RB, and a local measure, used to determine the goodness of the rules. We calculate these measures similarly to other HFRBCS methodologies focused on classification problems [21]. The global measure used is the accuracy per class, computed as:

$$Acc_i(X_i, RB) = \frac{|x_p \in X_i / FRM(x_p, RB) = Class(x_p)|}{|X_i|} \tag{16}$$

where $||$ is the number of patterns, X_i is the set of examples of the training set that belong to the ith class, $FRM(x_p,RB)$ is the class prediction of the pattern using the rules in the RB with the FRM used by the GP-COACH algorithm, and $Class(x_p)$ is the class label for example x_p. The local measure utilized is the accuracy for a rule, computed over the whole training set as

$$Acc(X, R_j) = \frac{|X^+(R_j)|}{|X(R_j)|} \tag{17}$$

It is important to remind that since we are using the normalized sum approach as FRM, $X^+(R_j)$ and $X(R_j)$ are defined as

- $X(R_j)$ is the set of examples that have a matching degree with the rule higher than 0 where this compatibility has contributed to classify the sample as the class label of the rule.
- $X^+(R_j)$ is the set of examples that have a matching degree with the rule higher than 0 where this compatibility has contributed to classify the sample as the class label of the rule and where the predicted class corresponds with the class label of the example.

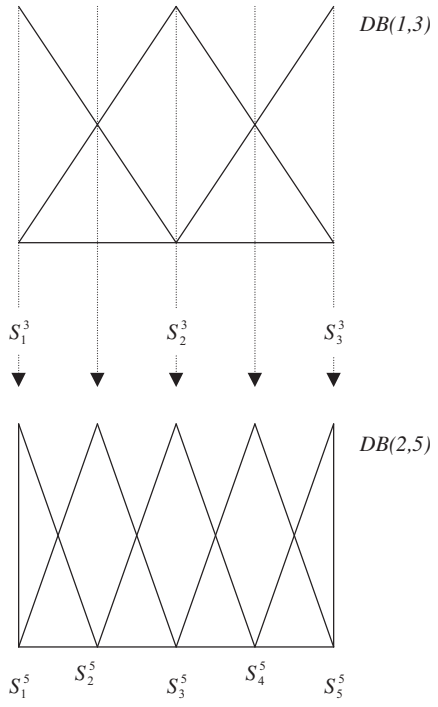


Fig. 6. Transition from a partition in DB(1,3) to another one in DB(2,5).

For each example in the training set, we obtain a set of rules that have contributed to the classification when we compute the global measure. Therefore, when we try to compute $X^+(R_j)$ and $X(R_j)$ we have for each rule the set of examples where the current rule has contributed to its classification.

Once we have computed the global measure and the local measure, we characterize the rules as *good* or *bad* according to the following calculation:

If $(Acc(X,R_j) \leqslant (1-\alpha)\cdot Acc_i(X_i,RB))$ Then
  R_j = *good rule*
Else
  R_j = *bad rule*

Good rules are kept in the rule population while *bad rules* are deleted from the current population. Then, new high granularity rules are created using as linguistic rule generator with the DB associated to layer $t+1$ and adopting as training set for this task a subset of the original training set including examples that meets some specified conditions. If after the generation of these rules we find repeated rules we only keep one copy of them, or if we find contradictory rules (rules with the same antecedent but with different consequents) we maintain the rule with a higher rule weight in the RB while the others are removed.

3.3.2. Integration of a HKB in the GP-COACH algorithm

The usage of a HKB in the inner way of running of the GP-COACH algorithm induces some changes in its structure. For example, the existence of the HRB which is composed by the RB of layer t and a RB of layer $t+1$ forces the GP-COACH algorithm to provide a mechanism to maintain these two RB levels. In our case, these RBs are merged and are evolved together in the different generations computed in the GP-COACH algorithm.

The rule population used in the algorithm is now a mixed population that combines primary rules and secondary rules where the secondary rules present different granularities. In this kind of population, genetic operators obtain rules according to the type of parent rule: primary rules obtain primary rules while secondary rules obtain secondary rules maintaining the granularity of the original rule. The only restriction in the application of the genetic operations appears in the usage of the crossover operation where the rules selected for the generation of a new rule must have the same granularity.

The global fitness score is modified to consider the different granularities of the rules in the population. The new global fitness function is:

$$\begin{aligned} global\_fitness = {} & w_1 \cdot accuracy + w_2 \cdot (1.0 - Var_N) + w_3 \\ & \cdot \left(1.0 - \frac{(Cond\_Low_N \cdot R\_Low + Cond\_High_N \cdot R\_High)}{R}\right) \\ & + w_4 \cdot (1.0 - Rul_N) \end{aligned} \tag{18}$$

where Var_N is the normalized average number of variables, $Cond\_Low_N$ is the normalized average number of conditions in low granularity rules, $Cond\_High_N$ is the normalized average number of conditions in high granularity rules, Rul_N is the normalized number of rules and R_Low,R_High,R are the number of low granularity rules, high granularity rules and total number of rules respectively.

To generate the high granularity rules some additional steps are performed just after the final step of a GP-COACH generation which is the construction of secondary rules for examples that have not been covered with the current rule base. This process is done performing the following operations:

1. The rules that compose the rule set are classified as *good rules* or *bad rules* as explained in the previous subsection.
2. *Good rules* are kept in the rule population and *bad rules* are directly deleted.
3. New high granularity rules are created using as linguistic rule generator the Chi et al. algorithm [10] with the DB associated to layer $t+1$ and adopting as training set for this task the examples that were classified by the rules that were considered *bad rules*.
4. Repeated and contradictory rules are searched for and only one copy of the best performing is kept.

Usually, when creating a hierarchical rule base, another step is added to improve the performance of the final model: a hierarchical rule selection step. In our case, since the hierarchical expansion of rules is embedded into each generation of the GP-COACH algorithm, adding a genetic selection process would increase considerably the run time of the approach. Therefore, this rule selection step is appended after the GP-COACH generations end combined with a tuning step to take advantage of the synergy between these refinements of the KB. Furthermore, GP-COACH tries to obtain a compact rule population with the token competition procedure making thus this delay of the rule selection step possible.

3.4. Hierarchical rule base selection and lateral tuning

In this last step, we analyze the use of genetic algorithms to select and tune a compact and cooperative set of fuzzy rules that obtain a high performance starting from the hierarchical rules generated in the previous step. In order to do so, we consider the approach used by Alcalá et al. [1] that uses the linguistic 2-tuples representation [32]. This representation allows the lateral displacement of the labels considering only one parameter (symbolic translation parameter), which involves a simplification of the tuning search space that aids the obtaining of optimal models. Particularly this happens when it is combined with a rule selection within the same process enabling it to take advantage of the positive synergy that both techniques present. In this way, this process for contextualizing the membership functions permits them to achieve a better covering degree while maintaining the original shapes, which results in accuracy improvements without a significant loss in the interpretability of the fuzzy labels. The symbolic translation parameter of a linguistic term is a number within the interval $[-0.5,0.5)$ that expresses the domain of a label when it is moving between its two lateral labels. Let us consider a set of labels S representing a fuzzy partition. Formally, we have the pair, $(s_i,\alpha_i), s_i \in S, \alpha_i \in [-0.5,0.5)$. An example is illustrated in Fig. 7 where we show the symbolic translation of a label represented by the pair $(S_2, -0.3)$.

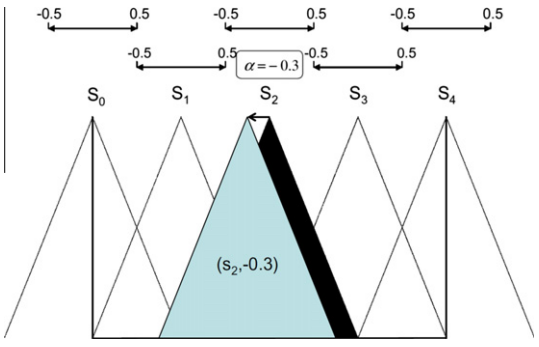


Fig. 7. Lateral displacement of a MF.

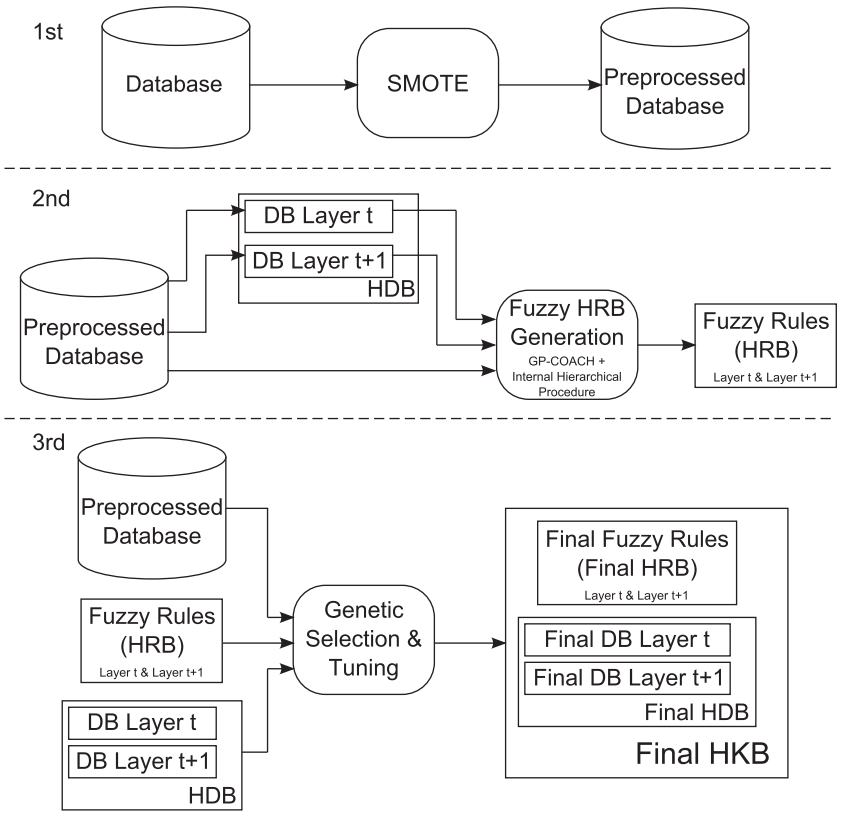


Fig. 8. Flowchart of GP-COACH-H.

Alcalá et al. [1] proposed two different rule representation approaches, a global approach and a local approach. In our algorithm, the tuning is applied to the level of linguistic partitions (global approach). In this way, the pair (X_i, label) takes the same tuning value in all the rules where it is considered. For example, X_1 - is (High, 0.3) will present the same value for those rules in which the pair "X_1 is High" was initially considered. This proposal decreases the tuning problem complexity, greatly easing the derivation of optimal models.

To accomplish this rule selection and lateral tuning process, we consider the use of a specific genetic algorithm, the CHC evolutionary algorithm [18] with the same scheme described in our previous works [21,22]. In the remainder of this section, we describe the specific features of our new tuning approach, which involves the codification of the solutions and initial gene pool, chromosome evaluation, crossover operator and restarting approach.

1. *Codification and Initial Gene Pool:* To combine the rule selection with the global lateral tuning, a double coding scheme for both rule selection (C_S) and lateral tuning (C_T) is used:
   - For the C_S part, each chromosome is a binary vector that determines when a rule is selected or not (alleles '1' and '0' respectively). Considering the M rules contained in the candidate rule set (rules from the two hierarchical levels considered), the corresponding part $C_S = \{c_1, \ldots, c_M\}$ represents a subset of rules composing the final rule base, so that, $If\, c_j = 1\, then (R_j \in RB) else (R_j \notin RB)$, with R_j being the corresponding jth rule in the candidate rule set and RB being the final RB.
   - For the C_T part, a real coding is considered. This part is the joint of the α parameters of each fuzzy partition. Let us consider the following number of labels per variable: (ml^1, ml^2, $\ldots$, ml^n) for low granularity rules and (mh^1, mh^2, $\ldots$, mh^n) for high granularity rules, with n being the number of system variables. Then, this part has the following form (where each gene is associated to the tuning value of the corresponding label): $C_T = (cl_{11}, \ldots, cl_{1ml^1}, cl_{21}, \ldots, cl_{2ml^2}, \ldots, cl_{n1}, \ldots, cl_{nml^n}, ch_{11}, \ldots, ch_{1mh^1}, ch_{21}, \ldots, ch_{2mh^2}, \ldots, ch_{n1}, \ldots, ch_{nmh^n})$.

Finally, a chromosome C is coded in the following way: $C = C_S C_T$. To make use of the available information, all the candidate rules are included in the population as an initial solution. To do this, the initial pool is obtained with the first individual having all genes with value '1' in the C_S part and all genes with value '0.0' in the C_T part. The remaining individuals are generated at random.

2. *Chromosome Evaluation:* To evaluate a determined chromosome we compute its accuracy over the training set. If two individuals obtain the same value, then the individual with the lower number of selected rules is preferred.
3. *Crossover Operator:* The crossover operator will depend on the chromosome part where it is applied:
   - In the C_S part, the half uniform crossover scheme (HUX) is employed.
   - For the C_T part, we consider the Parent Centric BLX (PCBLX) operator [31], which is based on BLX-α.
4. *Restarting Approach:* To get away from local optima, this algorithm uses a restart approach that is performed to improve the diversity of the population that may be reduced by the strong elitist pressure of the replacement scheme.

For details about the remainder features of the optimization process, please refer to Fernández et al. [21] and Fernández et al. [22].

3.5. Summary of the GP-COACH-H algorithm

Once every step of the algorithm has been explained we briefly sum up how the GP-COACH-H algorithm works. Fig. 8 depicts a flowchart of the GP-COACH-H algorithm.

Table 2
Parameter specification for the algorithms tested in the experimentation.

Algorithm	Parameters
FRBCS parameters	
GP-COACH and GP-COACH-H	Minimum t-norm, Maximum t-conorm, Rule Weight = Certainty Factor, Fuzzy Reasoning Method = Normalized Sum, Number of Fuzzy Labels (for basic GP-COACH) = 5 or 9, Number of Fuzzy Labels (for GP-COACH-H) = 5 for Low Granularity Rules and 9 for High Granularity Rules
HFRBCS(Chi)	Product t-norm, Rule Weight = Penalized Certainty Factor, Fuzzy Reasoning Method = Winning Rule, Number of Fuzzy Labels = 3 for Low Granularity Rules and 5 for High Granularity Rules
GP-COACH parameters	
GP-COACH and GP-COACH-H	Evaluations = 20000, Initial Population Size = 200, α (raw fitness) = 0.7, Crossover Probability = 0.5, Mutation Probability = 0.2, Dropping Condition Probability = 0.15, Insertion Probability = 0.15, Tournament size = 2, $w_1 = 0.8$, $w_2 = w_3 = 0.05$, $w_4 = 0.1$
Hierarchical procedure parameters	
GP-COACH-H and HFRBCS(Chi)	α (rule expansion) = 0.2, CHC Evaluations = 10,000, CHC Population Size = 61, CHC bits per gene (for GP-COACH-H) = 30
C4.5 parameters	
C4.5	Pruned=true, Confidence = 0.25 and Minimum number of item-sets per leaf = 2

There are three different steps in the building of the model:

1. *Preprocessing stage:* In this first step, GP-COACH-H preprocesses the original data-set to balance the class distribution. In order to do so, the SMOTE algorithm is used, as described in subSection 2.2.
2. *Generation of the HKB:* This stage is devoted to the generation of a two-layer HKB from the balanced data-set. This HKB is composed by two different DBs (each one with a different granularity level) and one RB that contains rules from the two hierarchies:
   (a) *HDB Generation*: The first layer DB is created with the same number of linguistic terms for all input variables, composed of symmetrical triangular-shaped and uniformly distributed membership functions. The second layer, is built preserving all the membership function modal points, corresponding to each linguistic term.
   (b) *HRB Generation:* In order to generate the HRB we use as a base the GP-COACH algorithm, which has been modified to incorporate in its internal way of running the creation of hierarchical rules. The adjustments reinforce the connection between the GP-COACH algorithm and the hierarchical methodology because they have been designed to get the greatest possible performance. Specifically, these modifications include:
      - A step to identify *good* and *bad* rules, where *bad* rules are deleted and the examples covered by them are used to create new high granularity rules.
      - Changes in the global fitness function considering the different granularities in the rule population.
      - A variation on the conditions of the application of the crossover operator where only rules with the same granularity level are allowed to produce an offspring.

   This HRB generation procedure uses the preprocessed data-set from the previous step and the membership functions defined by the HDB.
3. *Refinement of the HKB:* After the building of an initial HKB in the previous phase, another genetic procedure is applied to improve the final performance of this solution. In this step, rules that cooperate properly in the population are selected and the HDB is tuned with the 2-tuples linguistic representation. These optimizations are done in a single step to take advantage of the synergy that both techniques can achieve. The set of selected rules define the final HRB given as solution and the tuning parameters obtained modify the original HDB to create the final HDB which is the output of the algorithm.

4. Experimental framework

In this section, we present the set up of the experimental framework used to develop the analysis of our proposal. First we introduce the algorithms selected for the comparison with the proposed approach and their configuration parameters (subSection 4.1). Next, we provide details of the problems chosen for the experimentation (subSection 4.2). Finally, we present the statistical tests applied to compare the results obtained with the different classifiers (subSection 4.3).

4.1. Algorithms selected for the study and parameters

In order to test the performance of our approach, GP-COACH-H, several classification methods have been selected to perform the experimental study. These methods are:

- *GP-COACH* [7]: The original FRBCS that was used as base for our approach, a GP-based algorithm for the learning of compact and interpretable fuzzy rule bases that obtains good accuracy in high dimensional classification problems.
- *HFRBCS(Chi)* [21]: This approach obtains a Hierarchical Fuzzy Rule Base Classification System (HFRBCS) using the Chi et al. algorithm [10] as the linguistic rule generation method and has reported good results in imbalanced data-sets.
- *C4.5* [47]: A well-known decision tree which has shown a good behavior in the framework of imbalanced data-sets [6].

The configuration parameters used for these algorithms are shown in Table 2. All the methods were run using KEEL software[2] [3], following the default parameter values given in the KEEL platform to configure the methods, which were selected according to the recommendation of the corresponding authors of each algorithm, assuming that the choice of the values of the parameters was optimal.

Regarding the use of the SMOTE [9] and SMOTE+ENN [6] preprocessing methods, we consider only the 1-nearest neighbor (using the euclidean distance) to generate the synthetic samples, and we balance the training data to the 50% distribution. We only use SMOTE + ENN for C4.5 because it shows a positive synergy when pruning the tree [16].

4.2. Data-sets and data partitions

In order to analyze the quality of our approach GP-COACH-H against the algorithms introduced in the previous section, we have

[2] http://www.keel.es/.

Table 3
Summary of imbalanced data-sets.

Data-sets	#Ex.	#Atts.	Class (−; +)	%Class (−; +)	IR
ecoli034vs5	200	7	(p, imL, imU; om)	(10.00, 90.00)	9.00
yeast2vs4	514	8	(cyt; me2)	(9.92, 90.08)	9.08
ecoli067vs35	222	7	(cp, omL, pp; imL, om)	(9.91, 90.09)	9.09
ecoli0234vs5	202	7	(cp, imS, imL, imU; om)	(9.90, 90.10)	9.10
glass015vs2	172	9	(build-win-non_float-proc, tableware, build-win-float-proc; ve-win-float-proc)	(9.88, 90.12)	9.12
yeast0359vs78	506	8	(mit, me1, me3, erl; vac, pox)	(9.88, 90.12)	9.12
yeast02579vs368	1004	8	(mit, cyt, me3, vac, erl; me1, exc, pox)	(9.86, 90.14)	9.14
yeast0256vs3789	1004	8	(mit, cyt, me3, exc; me1, vac, pox, erl)	(9.86, 90.14)	9.14
ecoli046vs5	203	6	(cp, imU, omL; om)	(9.85, 90.15)	9.15
ecoli01vs235	244	7	(cp, im; imS, imL, om)	(9.83, 90.17)	9.17
ecoli0267vs35	224	7	(cp, imS, omL, pp; imL, om)	(9.82, 90.18)	9.18
glass04vs5	92	9	(build-win-float-proc, containers; tableware)	(9.78, 90.22)	9.22
ecoli0346vs5	205	7	(cp, imL, imU, omL; om)	(9.76, 90.24)	9.25
ecoli0347vs56	257	7	(cp, imL, imU, pp; om, omL)	(9.73, 90.27)	9.28
yeast05679vs4	528	8	(me2; mit, me3, exc, vac, erl)	(9.66, 90.34)	9.35
ecoli067vs5	220	6	(cp, omL, pp; om)	(9.09, 90.91)	10.00
vowel0	988	13	(hid; remainder)	(9.01, 90.99)	10.10
glass016vs2	192	9	(ve-win-float-proc; build-win-float-proc, build-win-non_float-proc,headlamps)	(8.89, 91.11)	10.29
glass2	214	9	(Ve-win-float-proc; remainder)	(8.78, 91.22)	10.39
ecoli0147vs2356	336	7	(cp, im, imU, pp; imS, imL, om, omL)	(8.63, 91.37)	10.59
led7digit02456789vs1	443	7	(0, 2, 4, 5, 6, 7, 8, 9; 1)	(8.35, 91.65)	10.97
glass06vs5	108	9	(build-win-float-proc, headlamps; tableware)	(8.33, 91.67)	11.00
ecoli01vs5	240	6	(cp, im; om)	(8.33, 91.67)	11.00
glass0146vs2	205	9	(build-win-float-proc, containers, headlamps, build-win-non_float-proc; ve-win-float-proc)	(8.29, 91.71)	11.06
ecoli0147vs56	332	6	(cp, im, imU, pp; om, omL)	(7.53, 92.47)	12.28
cleveland0vs4	177	13	(0; 4)	(7.34, 92.66)	12.62
ecoli0146vs5	280	6	(cp, im, imU, omL; om)	(7.14, 92.86)	13.00
ecoli4	336	7	(om; remainder)	(6.74, 93.26)	13.84
yeast1vs7	459	8	(nuc; vac)	(6.72, 93.28)	13.87
shuttle0vs4	1829	9	(Rad Flow; Bypass)	(6.72, 93.28)	13.87
glass4	214	9	(containers; remainder)	(6.07, 93.93)	15.47
page-blocks13vs2	472	10	(graphic; horiz.line, picture)	(5.93, 94.07)	15.85
abalone9vs18	731	8	(18; 9)	(5.65, 94.25)	16.68
glass016vs5	184	9	(tableware; build-win-float-proc, build-win-non_float-proc,headlamps)	(4.89, 95.11)	19.44
shuttle2vs4	129	9	(Fpv Open; Bypass)	(4.65, 95.35)	20.5
yeast1458vs7	693	8	(vac; nuc, me2, me3, pox)	(4.33, 95.67)	22.10
glass5	214	9	(tableware; remainder)	(4.20, 95.80)	22.81
yeast2vs8	482	8	(pox; cyt)	(4.15, 95.85)	23.10
yeast4	1484	8	(me2; remainder)	(3.43, 96.57)	28.41
yeast1289vs7	947	8	(vac; nuc, cyt, pox, erl)	(3.17, 96.83)	30.56
yeast5	1484	8	(me1; remainder)	(2.96, 97.04)	32.78
ecoli0137vs26	281	7	(pp, imL; cp, im, imU, imS)	(2.49, 97.51)	39.15
yeast6	1484	8	(exc; remainder)	(2.49, 97.51)	39.15
abalone19	4174	8	(19; remainder)	(0.77, 99.23)	128.87

selected several highly imbalanced and borderline imbalanced data-sets.

Specifically, as highly imbalanced data-sets, we have selected 44 data-sets from KEEL data-set repository[3] [2] with an imbalance ratio (IR) [46] greater than 9. The data are summarized in Table 3, where we denote the number of examples (#Ex.), number of attributes (#Atts.), class name of each class (positive and negative), class attribute distribution and IR. This table is in ascending order according to the IR.

Inspired by Kubat and Matwin [40], Napierala et al. [44] created several artificial data-sets that contain borderline examples in an imbalanced scenario to address the correct identification of those examples. These data-sets have three different shapes of the positive class: *subclus* (Fig. 9), *clover* (Fig. 10) and *paw* (Fig. 11), all surrounded uniformly by the negative class. For each shape, we have data-sets from two different sizes and IR: data-sets with 600 examples with an IR of 5 and data-sets with 800 examples with an IR of 7. Each one of these data-sets is affected by different disturbance ratio levels (0%, 30%, 50%, 60% and 70%). The disturbance ratio is simulated increasing the ratio of borderline examples from the positive class subregions.

To develop the different experiments we consider a *5-fold cross-validation model*, i.e., five random partitions of data with a 20% and the combination of 4 of them (80%) as training and the remaining ones as test. For each data-set we consider the average results of the five partitions. The data-sets used in this study use the partitions provided by the KEEL data-set repository in the imbalanced classification data-set section.[4]

4.3. Statistical tests for performance comparison

Statistical analysis needs to be carried out in order to find significant differences among the results obtained by the studied methods [24]. We consider the use of non-parametric tests, according to the recommendations made in [25,24] where a set of simple, safe and robust non-parametric tests for statistical comparisons of classifiers is presented. These tests are used due to the fact that the initial conditions that guarantee the reliability of the parametric tests may not be satisfied, causing the statistical analysis to lose credibility [50].

The Wilcoxon test [50] will be used as a non-parametric statistical procedure in order to conduct pairwise comparisons between two algorithms. For multiple comparisons we use the Iman-Davenport

[3] http://www.keel.es/datasets.php.

[4] http://www.keel.es/imbalanced.php.

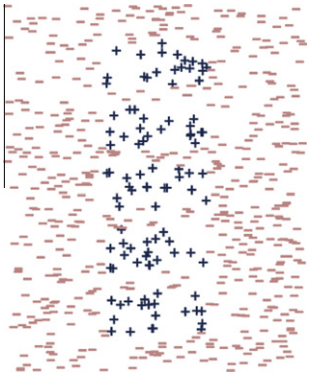


Fig. 9. Subclus.

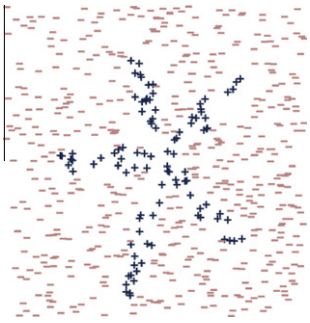


Fig. 10. Clover.

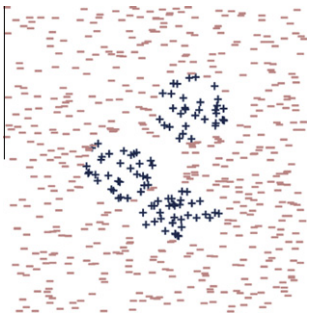


Fig. 11. Paw.

test to detect statistical differences among a group of results, and the Holm post-hoc test in order to find which algorithms are distinctive among a 1 × n comparison.

The post-hoc procedure allows us to know whether a hypothesis of comparison of means could be rejected at a specified level of significance α. However, it is very interesting to compute the p-value associated with each comparison, which represents the lowest level of significance of a hypothesis that results in a rejection. It is the adjusted p-value. In this manner, we can know whether two algorithms are significantly different and how different they are.

Furthermore, we consider the average ranking of the algorithms, in order to show how good a method is with respect to its partners. This ranking is obtained by assigning a position to each algorithm depending on its performance for each data-set. The algorithm which achieves the best accuracy in a specific data-set will have the first ranking (value 1); then, the algorithm with the second best accuracy is assigned rank 2, and so forth. This task is carried out for all data-sets and finally an average ranking is computed as the mean value of all rankings.

These tests are suggested in the studies presented in [25,24], where their use in the field of machine learning is highly recommended. For a wider description of the use of these tests, please refer to the website on *Statistical Inference in Computational Intelligence and Data Mining*.[5]

5. Experimental study

In this section, we present a set of experiments to illustrate and demonstrate the behavior of GP-COACH-H. These experiments are designed towards two objectives: to exemplify how the GP-COACH-H algorithm works, and to determine its robustness for highly and borderline imbalanced data-sets.

We organize those experiments in the following way. First, Section 5.1 presents a case of study over one one of the highly imbalanced data-sets presented in the previous section. Next, Section 5.2 contains an analysis of the impact of the hierarchical step in the algorithm. Section 5.3 studies the the importance of the usage of a preprocessing step when dealing with highly imbalanced data-sets. Later, Section 5.4 performs a global comparison among the fuzzy classification methods and C4.5 over the highly imbalanced data-sets. Finally, in Section 5.5, this global comparison is also carried out over the borderline imbalanced data-sets.

5.1. Sample procedure of the GP-COACH-H algorithm: a case of study

In order to illustrate how GP-COACH-H works we have selected the *glass0146vs2* data-set. We will follow the algorithm operations and the results it provides. The *glass0146vs2* data-set is a highly imbalanced data-set from the KEEL data-set repository,[6] with 9 input attributes, 205 instances and an IR equal to 11.06. We have selected this data-set as one with a small size whose results can be easily interpreted.

For this specific run, we have chosen the 3rd partition from the 5-fcv used in all the experiments. This partition uses 164 instances for training (14 positive and 150 negative) and 41 for test (3 positive and 38 negative), using the 9 input attributes of the whole data-set. The first step of the GP-COACH-H algorithm (see Fig. 8) uses the SMOTE algorithm to balance the class distribution. Therefore, we apply the SMOTE algorithm and we obtain a new training set that contains 300 instances, 150 instances for each class.

The second step starts using the preprocessed data-set to generate the HKB. In order to generate the HKB, we first generate the HDB from the available data. The HDB is generated (as was explained in the previous sections) with the same number of linguistic terms for all input variables, composed of symmetrical triangular-shaped and uniformly distributed membership functions. The second layer, is built preserving all the membership function modal points, corresponding to each linguistic term. Figs. 12 and 13 show the linguistic variables generated for the *Mg* attribute, according to the given instructions.

Once we have generated the HDB, we start the GP procedure to generate the HRB. This procedure evolves a rule population through several generations, including the usage of genetic operators to generate new individuals, the token competition procedure to delete irrelevant rules and the hierarchical creation of new rules in each step. At the end of the iterations, a rule base with different granularity rules is obtained. In Fig. 14, the rules generated using the generated HDB and the preprocessed training set are shown.

At this point, we start the last step of the algorithm which is the genetic rule selection and lateral tuning of the variables. To obtain the final solution, we use the preprocessed set from the first step and the HKB generated previously. The genetic search looks for a

[5] http://sci2s.ugr.es/sicidm/.
[6] http://www.keel.es/imbalanced.php.

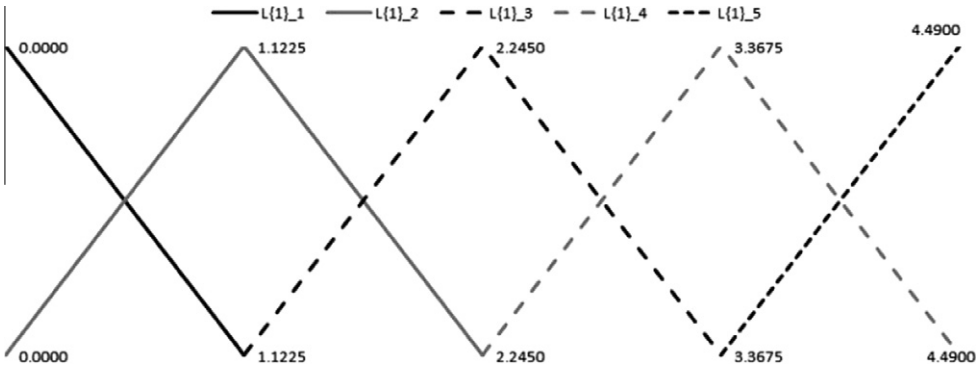


Fig. 12. Database Layer 1 with 5 labels, M_g attribute.

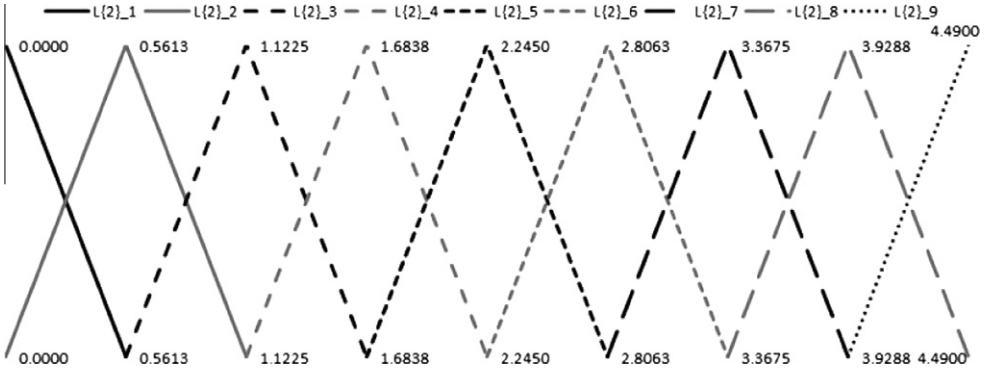


Fig. 13. Database Layer 2 with 9 labels, M_g attribute.

```
@Number of rules: 13

1{1}: IF Mg IS (L{1}_1 OR L{1}_2 OR L{1}_3) THEN negative with RW: .9969
2{1}: IF Ca IS (L{1}_1 OR L{1}_4 OR L{1}_5) THEN negative with RW: .9999
3{1}: IF RI IS (L{1}_4 OR L{1}_5) THEN negative with RW: 1.0
4{1}: IF Al IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Fe IS (L{1}_3 OR L{1}_5) THEN negative with RW: 1.0
5{1}: IF Al IS L{1}_3 AND Si IS (L{1}_1 OR L{1}_2) AND Fe IS (L{1}_2 OR L{1}_5) THEN negative with RW: 1.0
6{1}: IF Na IS (L{1}_1 OR L{1}_2 OR L{1}_5) AND Si IS (L{1}_1 OR L{1}_2 OR L{1}_5) THEN negative with RW: .9969
7{1}: IF RI IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Al IS L{1}_3 AND Fe IS (L{1}_2 OR L{1}_5) THEN negative with RW:
      .9904
8{1}: IF RI IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Na IS (L{1}_1 OR L{1}_2 OR L{1}_5) AND Fe IS (L{1}_2 OR L{1}_5)
      THEN negative with RW: .9590
9{2}: IF RI IS L{1}_3 AND Na IS L{1}_4 AND Mg IS L{1}_4 AND Al IS L{1}_2 AND Si IS L{1}_3 AND K IS L{1}_1 AND
      Ca IS L{1}_2 AND Ba IS L{1}_1 AND Fe IS L{1}_2 THEN positive with RW: .8156
10{2}: IF RI IS L{1}_2 AND Na IS L{1}_3 AND Mg IS L{1}_4 AND Al IS L{1}_2 AND Si IS L{1}_3 AND K IS L{1}_1 AND
       Ca IS L{1}_2 AND Ba IS L{1}_1 AND Fe IS L{1}_1 THEN positive with RW: .6675
11{2}: IF RI IS L{2}_3 AND Na IS L{2}_5 AND Mg IS L{2}_7 AND Al IS L{2}_2 AND Si IS L{2}_6 AND K IS L{2}_1 AND
       Ca IS L{2}_3 AND Ba IS L{2}_1 AND Fe IS L{2}_1 THEN positive with RW: .9654
12{2}: IF RI IS L{2}_4 AND Na IS L{2}_5 AND Mg IS L{2}_7 AND Al IS L{2}_2 AND Si IS L{2}_5 AND K IS L{2}_1 AND
       Ca IS L{2}_4 AND Ba IS L{2}_1 AND Fe IS L{2}_2 THEN positive with RW: .7443
13{2}: IF RI IS L{2}_5 AND Na IS L{2}_7 AND Mg IS L{2}_1 AND Al IS L{2}_3 AND Si IS L{2}_5 AND K IS L{2}_1 AND
       Ca IS L{2}_6 AND Ba IS L{2}_1 AND Fe IS L{2}_2 THEN negative with RW: 1.0
```

Fig. 14. Rules generated after the Fuzzy HRB Generation.

new HKB that better represents the data. Figs. 15–17 show the new HDB and HRB obtained, which are the final output of the GP-COACH-H algorithm.

5.2. Analysis of the impact of the hierarchical levels over the imbalanced data-sets

This subsection is devoted to the impact of the usage of the HKB in the GP-COACH-H algorithm in relation to not using a HKB and use a traditional KB instead. In this manner, we will detect the influence of this component of the GP-COACH-H algorithm thus justifying its use.

We will compare the results of the GP-COACH-H algorithm according to the fuzzy HKB generated after the application of the GP procedure to the results of the basic GP-COACH algorithm with 5 and 9 labels, using SMOTE as preprocessing algorithm in both cases. The performance measures used are sensitivity and specificity to observe the impact for each class. Table 4 shows the average results for each algorithm over the highly imbalanced data-sets. The complete table of results for all data-sets can be found in the appendix of this work.

Considering the sensitivity measure the best performing average algorithm is the basic GP-COACH with 5 labels, however, if we look at the specificity measure then the best performing algorithm is the basic GP-COACH with 9 labels. Therefore, we need to consider the effectiveness for each class separately.

Contemplating the positive class, we can observe that the best performance in training is higher for the hierarchical version, being

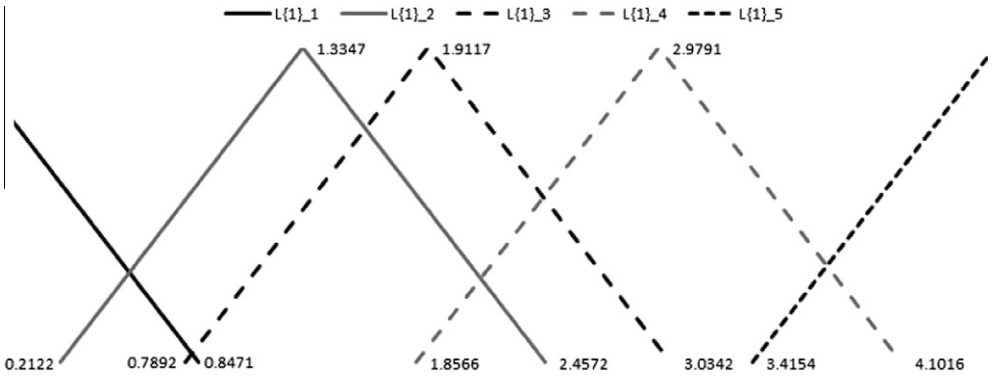


Fig. 15. Final database Layer 1 with 5 labels, M_g attribute.

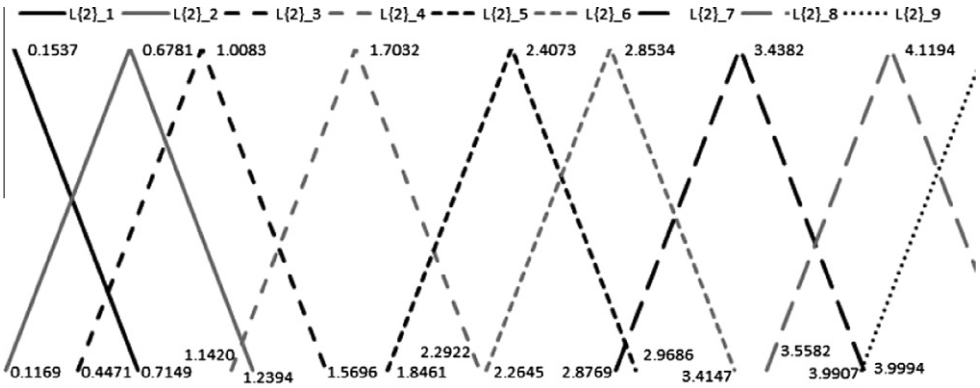


Fig. 16. Final database Layer 2 with 9 labels, M_g attribute.

Table 4
Average results for GP-COACH-5, GP-COACH-9 and GP-COACH-H for the highly imbalanced data-sets.

Data-set	Sensitivity$_{tr}$	Sensitivity$_{tst}$	Specificity$_{tr}$	Specificity$_{tst}$
GP-COACH-5	.9097 ± .0307	**.7809 ± .1212**	.8643 ± .0307	.8531 ± .1212
GP-COACH-9	.8983 ± .0267	.7319 ± .1334	.9231 ± .0267	**.9055 ± .1334**
GP-COACH-H	.9398 ± .0204	.7797 ± .1233	.9025 ± .0204	.8855 ± .1233

Table 5
Average results for GP-COACH versions with and without SMOTE preprocessing for the highly imbalanced data-sets.

Data-set	No preprocessing		SMOTE preprocessing	
	GM$_{tr}$	GM$_{tst}$	GM$_{tr}$	GM$_{tst}$
GP-COACH-5	.4789 ± .1017	.3677 ± .1922	.8763 ± .0307	**.7897 ± .1212**
GP-COACH-9	.5074 ± .0871	.3929 ± .1996	.9056 ± .0267	**.7845 ± .1334**
GP-COACH-H	.4536 ± .1216	.3439 ± .1697	.9576 ± .0121	**.8175 ± .1193**

able to describe the training set more accurately than in the presence of low granularity rules only. Therefore, our initial intuition where the HKB was able to better describe difficult data spaces is confirmed. Comparing the training results in relation to the test results we notice a drop in performance for all the algorithms where GP-COACH-5 gets the best results, GP-COACH-H obtains similar results to GP-COACH-5 and GP-COACH-9 accomplishes lower results than the other two.

Analyzing the results associated to the negative class, we see an almost opposite situation. For training results the GP-COACH-9 algorithm is the algorithm that best describes the data, a situation where GP-COACH-H is supposed to be found. Nevertheless, GP-COACH-H is designed to specifically deal with imbalanced data-sets concentrating on the positive class so is logical that it does not characterize the negative class as well as the previous case. Confronting the training results with the test results we find a drop in the performance on equal levels for each approaches. Therefore, GP-COACH-9 is the best performing algorithm for the negative class, closely followed by GP-COACH-H where GP-COACH-5 performance falls behind those two approaches.

After checking the performance in each class, we discover that the basic GP-COACH is a powerful tool to describe one of our classes depending on the number of labels used. Nevertheless, if we choose a specific number of labels to focus on one class the final performance is degraded in the other one. Consequently, the GP-COACH-H approach that combines low granularity and high granularity rules is able to address the description of both classes accordingly. Its performance does not exceed the results of the basic algorithm, however, it goes closely after them in each class. Furthermore, there is not a high decrease in performance for the class

```
@Number of rules: 10

1{1}: IF Mg IS (L{1}_1 OR L{1}_2 OR L{1}_3) THEN negative with RW: 1.0
2{1}: IF Ca IS (L{1}_1 OR L{1}_4 OR L{1}_5) THEN negative with RW: .9871
3{1}: IF RI IS (L{1}_4 OR L{1}_5) THEN negative with RW: .9981
4{1}: IF Al IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Fe IS (L{1}_3 OR L{1}_5) THEN negative with RW: .9892
5{1}: IF Al IS L{1}_3 AND Si IS (L{1}_1 OR L{1}_2) AND Fe IS (L{1}_2 OR L{1}_5) THEN negative with RW: .9902
6{1}: IF Na IS (L{1}_1 OR L{1}_2 OR L{1}_5) AND Si IS (L{1}_1 OR L{1}_2 OR L{1}_5) THEN negative with RW: 1.0
7{1}: IF RI IS (L{1}_3 OR L{1}_4 OR L{1}_5) AND Na IS (L{1}_1 OR L{1}_2 OR L{1}_5) AND Fe IS (L{1}_2 OR L{1}_5)
      THEN negative with RW: .9461
8{2}: IF RI IS L{1}_3 AND Na IS L{1}_4 AND Mg IS L{1}_4 AND Al IS L{1}_2 AND Si IS L{1}_3 AND K IS L{1}_1 AND Ca
      IS L{1}_2 AND Ba IS L{1}_1 AND Fe IS L{1}_2 THEN positive with RW: .6544
9{2}: IF RI IS L{1}_2 AND Na IS L{1}_3 AND Mg IS L{1}_4 AND Al IS L{1}_2 AND Si IS L{1}_3 AND K IS L{1}_1 AND Ca
      IS L{1}_2 AND Ba IS L{1}_1 AND Fe IS L{1}_1 THEN positive with RW: .6719
10{2}: IF RI IS L{2}_3 AND Na IS L{2}_5 AND Mg IS L{2}_7 AND Al IS L{2}_2 AND Si IS L{2}_6 AND K IS L{2}_1 AND Ca
       IS L{2}_3 AND Ba IS L{2}_1 AND Fe IS L{2}_1 THEN positive with RW: .9561
```

Fig. 17. Final rules generated with the GP-COACH-H algorithm.

Table 6
Average results for FRBCS methods and C4.5 for the highly imbalanced data-sets. SMOTE preprocessing for FRBCS methods, SMOTE+ENN for C4.5.

Data-set	GM_{tr}	GM_{tst}
GP-COACH-5	.8763 ± .0307	.7897 ± .1212
GP-COACH-9	.9056 ± .0267	.7845 ± .1334
HFRBCS(Chi)	.9331 ± .0117	.7901 ± .1325
GP-COACH-H	.9576 ± .0121	**.8175 ± .1193**
C4.5	.9549 ± .0180	.7848 ± .1452

Table 7
Average rankings and adjusted p-values using Holm's post-hoc procedure for FRBCS methods and C4.5 adopting the GM measure for the highly imbalanced data-sets.

Algorithm	Average ranking	Adjusted p-value (Holm's test)
GP-COACH-H	2.4091	
GP-COACH-9	3.0227	0.0862
GP-COACH-5	3.0909	0.0862
C4.5	3.2045	0.0549
HFRBCS(Chi)	3.2727	0.0416

Table 8
Average results for FRBCS methods and C4.5 for the borderline imbalanced data-sets. SMOTE preprocessing for FRBCS methods, SMOTE+ENN for C4.5.

Data-set	GM_{tr}	GM_{tst}
GP-COACH-5	.7899 ± .0218	.7630 ± .0578
GP-COACH-9	.8103 ± .0330	.7628 ± .0705
HFRBCS(Chi)	.8316 ± .0195	.7992 ± .0461
GP-COACH-H	.8674 ± .0157	**.8234 ± .0428**
C4.5	.8881 ± .0244	.8208 ± .0462

Table 9
Average rankings and adjusted p-values using Holm's post-hoc procedure for FRBCS methods and C4.5 adopting the GM measure for the borderline imbalanced data-sets.

Algorithm	Average ranking	Adjusted p-value (Holm's test)
GP-COACH-H	1.7333	
C4.5	1.9000	0.6831
HFRBCS(Chi)	3.0667	0.0022
GP-COACH-9	3.8667	0.0000
GP-COACH-5	4.4333	0.0000

Table 10
Wilcoxon test to compare GP-COACH-H against C4.5 in borderline imbalanced data-sets. R^+ corresponds to the sum of the ranks for GP-COACH-H and R^- to C4.5.

Comparison	R^+	R^-	p-Value
GP-COACH-H vs C4.5	261.0	204.0	0.551

as in the basic algorithm. In this manner, GP-COACH-H is able to profit from the descriptive power of each granularity level obtaining a good balance between the performance of both classes.

5.3. Analysis of the suitability of the preprocessing step for imbalanced problems

In this part of the study, our aim is to show the suitability of the preprocessing step included in GP-COACH-H as the first step of the algorithm. We also check the performance of applying this preprocessing step to the basic GP-COACH algorithm in order to show the necessity of this procedure when dealing with imbalanced data-sets, thus justifying the inclusion of this step in our proposal.

According to this objective, we show the average GM results in training and test in Table 5, together with the corresponding standard deviation, for the basic GP-COACH algorithm and for the hierarchical GP-COACH-H with and without SMOTE preprocessing over the highly imbalanced data-sets presented in Section 4.2. The complete table of results for all data-sets is shown in the appendix of this work. We observe that the best result in test (which is stressed in boldface) always corresponds to the one obtained when the SMOTE preprocessing is applied. Furthermore, there is an enormous difference between the usage or not usage of preprocessing. Therefore, we conclude that the usage of SMOTE as preprocessing clearly outperforms the usage of the original data-sets making the use of this methodology a necessity in the framework of imbalanced data-sets.

5.4. Analysis of GP-COACH-H on highly imbalanced data-sets

The following part of the study will consider the performance of the GP-COACH-H algorithm in contrast with other FRBCS learning proposals and with the C4.5 algorithm. Table 6 shows the average GM results in training and test together with the corresponding standard deviation for the highly imbalanced data-sets considered. By rows, we can observe the results for the basic GP-COACH method with 5 and 9 labels (GP-COACH-5 and GP-COACH-9), the HFRBCS(Chi), the proposed GP-COACH-H and the C4.5 decision tree. The best average case in test is highlighted in bold. The complete table of results for all data-sets is also shown in the appendix of this work together with the results of the previous experiments. We remind that SMOTE is used for the FRBCS whereas SMOTE+ENN is applied in conjunction with C4.5 along all the experiments.

According to the average values shown in this table the best method in highly imbalanced data-sets is the GP-COACH-H. To carry out the statistical study we first check for significant differences among the algorithms using an Iman-Davenport test. The p-value (0.0779) is low enough to reject the null equality hypothesis with a high confidence level. Thus, we can conclude that significant differences do exist, proceeding by showing in Table 7 the average

ranks of the algorithms and the adjusted p-values computed by the Holm test. Looking at this table we can notice that GP-COACH-H obtains the lower ranking which makes it the control method used for the post-hoc computation. As all the adjusted p-values are sufficiently low to reject the null-hypothesis in all cases, the assumption where GP-COACH-H is the best performing method considered for highly imbalanced data-sets is reinforced.

5.5. Analysis of GP-COACH-H on borderline imbalanced data-sets

In the last part of our study, we want to analyze the behavior of the GP-COACH-H proposal in the scenario of imbalance borderline data-sets. We will take into account the same algorithms considered in the analysis for highly imbalanced data-sets, namely, the basic GP-COACH method with 5 and 9 labels (GP-COACH-5 and GP-COACH-9), HFRBCS(Chi), GP-COACH-H and the C4.5 decision tree. Table 8 shows the average results in training and test together with the corresponding standard deviation for the algorithms used in the study over the borderline imbalanced data-sets. As in previous tables, the best average case in test is highlighted in bold and the complete table of results for the borderline imbalanced data-sets is also shown in the appendix of this work.

Observing the average results table we detect GP-COACH-H as the method with the best average results. Similarly to the procedure used in the highly imbalanced data-sets comparison we start the statistical study for borderline imbalanced data-sets computing the Iman-Davenport test to discern if there are significant differences among the algorithms. The p-value computed is zero, implying that there are differences between the algorithms. Therefore, we perform the Holm test as post-hoc procedure. Table 9 contains the ranks of the algorithms and the adjusted p-values computed using the Holm test.

According to Table 9 the lowest ranking corresponds to GP-COACH-H turning it into the control method used in the Holm test as the best performing method for borderline data-sets. In this case, the adjusted p-values associated to the basic GP-COACH (with 5 and 9 labels) and to HFRBCS(Chi) are low enough to reject the null-hypothesis with a high confidence level. This means, that our proposal GP-COACH-H is the best performing FRBCS in borderline imbalanced data-sets. In the remaining case (C4.5), we perform a Wilcoxon test (Table 10) in order to check if we find differences between both algorithms.

In this case, the p-value computed does not reject the null hypothesis. Nevertheless, GP-COACH-H achieves a higher sum of ranks, which means that GP-COACH-H has obtained a greater performance in a superior number of data-sets than C4.5, turning GP-COACH-H into a competitive method. Furthermore, the average performance of GP-COACH-H is better than the performance of C4.5 and the standard deviation is lower which causes GP-COACH-H to be a more robust method in each occasion.

To sum up, our experimental study has shown that GP-COACH-H is an algorithm that presents a good behavior in the framework of imbalanced data-sets, specifically, when dealing with high imbalanced data and borderline imbalanced data. The design of GP-COACH-H integrates different strategies to deal with the problem that help to overcome the difficulties when they appear. Specifically, the preprocessing step is used to counter the imbalance problem, the hierarchical procedure is added to the FRBCS used as base to obtain a better representation of the data-set in difficult areas such as small disjuncts or borderline samples and the rule selection combined with tuning refines the results obtained improving the overall results. These schemes combined together deal with the mentioned problems in conjunction generating good results.

6. Concluding remarks

In this paper we have presented a FRBCS with different granulation levels that integrates rule selection and the 2-tuples tuning approach to improve the performance in imbalanced data-sets. The proposal integrates data sampling together with algorithm modifications to the basic approach and adapts its behavior to the different granulation levels considered, adding a post-processing step that helps the hierarchical fuzzy rule base classification system to have a better adaptation to the context of each problem and therefore to enhance its global behavior.

The proposed hierarchical fuzzy rule based classification was compared to the GP-COACH algorithm, HFRBCS algorithm and the C4.5 decision tree in order to demonstrate its good performance. The experimental study justifies the combination of SMOTE with the algorithmic modifications such as the usage of a hierarchical knowledge base in order to increase the performance in the imbalanced data-set scenario. Moreover, the results obtained when we deal with this scenario evidence the interest of this proposal. Specifically, this proposal outperforms the other approaches in the framework of highly imbalanced data-sets, which usually is an scenario where most algorithms have lots of difficulties to perform properly.

For borderline imbalanced data-sets our approach shows a better behavior than other FRBCSs used in the experimental studio and maintains a competitive performance when it is compared with C4.5. These results have been contrasted by several non-parametric statistical procedures that reinforce the extracted conclusions.

As future work, we consider several lines of work centered on the features of GP-COACH-H that can still be enhanced to obtain a better performance. One possibility includes the modification of the genetic operations to achieve a multi-objective procedure that enables a trade-off between interpretability and accuracy. Moreover, we want to study in depth the data intrinsic characteristics that hinder the performance in imbalanced data-sets and incorporate this knowledge into the model with a specialized strategy for each case. Another possibility focus on the balance level of the preprocessing step. If an equal balance is not needed and can be substituted by a lower number of instances then the run time of the algorithm will decrease.

Acknowledgments

This work has been supported by the Spanish Ministry of Science and Technology under Projects TIN2011-28488 and TIN2008-06681-C06-02, and the Andalusian Research Plan P10-TIC-6858 and TIC-3928. V. López holds a FPU scholarship from Spanish Ministry of Education.

Appendix A. Detailed results for the experimental study

In this appendix we present the complete results tables for all the algorithms used in this work. Thus, the reader can observe the full training and test results, with their associated standard deviation, in order to compare the performance of each approach. In Table 11 we show the detailed results for the GP-COACH-5, GP-COACH-9 and GP-COACH-H versions with SMOTE preprocessing for the GP procedure using the specificity and sensitivity measures. Next, in Table 12 we show the results for the basic GP-COACH method and the hierarchical GP-COACH-H with and without SMOTE preprocessing. Later, the results for each FRBCS method with SMOTE preprocessing and C4.5 with SMOTE+ENN preprocessing over the highly imbalanced data-sets are shown in Table 13. Finally, Table 14 presents the results for the same algorithms as Table 13 over the borderline data-sets considered.

Table 11
Complete table of results for GP-COACH-5, GP-COACH-9 and GP-COACH-H after the GP procedure using the specificity and sensitivity measures.

Data-set	GP-COACH-5				GP-COACH-9				GP-COACH-H			
	$Sensitivity_{tr}$	$Sensitivity_{tst}$	$Specificity_{tr}$	$Specificity_{tst}$	$Sensitivity_{tr}$	$Sensitivity_{tst}$	$Specificity_{tr}$	$Specificity_{tst}$	$Sensitivity_{tr}$	$Sensitivity_{tst}$	$Specificity_{tr}$	$Specificity_{tst}$
ecoli034vs5	.9750 ± .0165	.8500 ± .1282	.9764 ± .0165	.9722 ± .1282	.9875 ± .0158	.9000 ± .0709	.9653 ± .0158	.9556 ± .0709	.9375 ± .0567	.8500 ± .0759	.9806 ± .0567	.9667 ± .0759
yeast2vs4	.9316 ± .0092	.8818 ± .0412	.9196 ± .0092	.9179 ± .0412	.9265 ± .0044	.8818 ± .0381	.9319 ± .0044	.9288 ± .0381	.9365 ± .0166	.8636 ± .0471	.9352 ± .0166	.9308 ± .0471
ecoli067vs35	.9654 ± .0193	.8400 ± .2093	.9200 ± .0193	.8650 ± .2093	.9438 ± .0210	.8400 ± .2265	.9412 ± .0210	.9250 ± .2265	.9660 ± .0292	.8400 ± .2248	.9463 ± .0292	.9200 ± .2248
ecoli0234vs5	.9625 ± .0095	.7500 ± .1552	.9794 ± .0095	.9338 ± .1552	.9625 ± .0409	.8000 ± .1648	.9670 ± .0409	.9174 ± .1648	.9875 ± .0111	.8500 ± .1239	.9822 ± .0111	.9392 ± .1239
glass015vs2	.7429 ± .1337	.4833 ± .2183	.4774 ± .1337	.4968 ± .2183	.8077 ± .0511	.1833 ± .3032	.8581 ± .0511	.7677 ± .3032	.8978 ± .0440	.6000 ± .0739	.7742 ± .0440	.7677 ± .0739
yeast0359vs78	.7650 ± .0840	.6400 ± .1244	.5273 ± .0840	.5312 ± .1244	.3700 ± .0258	.3600 ± .0833	.8499 ± .0258	.8418 ± .0833	.8450 ± .0199	.7000 ± .0820	.8164 ± .0199	.8026 ± .0820
yeast02579vs368	.8763 ± .0093	.8700 ± .0376	.9577 ± .0093	.9514 ± .0376	.8864 ± .0109	.8900 ± .0395	.9279 ± .0109	.9204 ± .0395	.8788 ± .0093	.8600 ± .0488	.9577 ± .0093	.9547 ± .0488
yeast0256vs3789	.7096 ± .0136	.6858 ± .0676	.9251 ± .0136	.9271 ± .0676	.7322 ± .0149	.7063 ± .0563	.9022 ± .0149	.8994 ± .0563	.7247 ± .0154	.7063 ± .0598	.9191 ± .0154	.9182 ± .0598
ecoli046vs5	.9750 ± .0168	.9000 ± .1248	.9740 ± .0168	.9509 ± .1248	.9875 ± .0174	.8500 ± .2166	.9836 ± .0174	.9566 ± .2166	1.0000 ± .0073	.8500 ± .2117	.9727 ± .0073	.9401 ± .2117
ecoli01vs235	.9689 ± .0151	.8600 ± .1131	.9125 ± .0151	.8955 ± .1131	.9689 ± .0152	.9100 ± .0670	.9398 ± .0152	.9227 ± .0670	.9479 ± .0184	.7700 ± .1915	.9443 ± .0184	.9364 ± .1915
ecoli0267vs35	.9216 ± .0211	.8000 ± .1311	.9209 ± .0211	.9156 ± .1311	.9778 ± .0334	.8000 ± .1125	.9220 ± .0334	.8916 ± .1125	.9444 ± .0260	.8000 ± .0928	.9073 ± .0260	.8709 ± .0928
glass04vs5	1.0000 ± .0208	.9000 ± .1277	.9338 ± .0208	.9287 ± .1277	1.0000 ± .0247	1.0000 ± .0134	.9426 ± .0247	.9279 ± .0134	1.0000 ± .0365	.8000 ± .4020	.9190 ± .0365	.8412 ± .4020
ecoli0346vs5	1.0000 ± .0028	.8000 ± .1132	.9919 ± .0028	.9784 ± .1132	.9875 ± .0159	.8500 ± .0632	.9811 ± .0159	.9459 ± .0632	1.0000 ± .0107	.8500 ± .0608	.9770 ± .0107	.9622 ± .0608
ecoli0347vs56	.9700 ± .0104	.8000 ± .1039	.9461 ± .0104	.9224 ± .1039	.9700 ± .0166	.8000 ± .1459	.9590 ± .0166	.9397 ± .1459	.9800 ± .0069	.8400 ± .0923	.9483 ± .0069	.9309 ± .0923
yeast05679vs4	.7843 ± .0198	.7836 ± .0759	.8569 ± .0198	.8490 ± .0759	.7990 ± .0158	.7636 ± .0858	.8753 ± .0158	.8616 ± .0858	.8184 ± .0117	.7255 ± .0653	.8496 ± .0117	.8449 ± .0653
ecoli067vs5	1.0000 ± .0144	.8000 ± .1277	.9113 ± .0144	.8850 ± .1277	.9625 ± .0162	.8500 ± .0884	.9488 ± .0162	.9400 ± .0884	.9625 ± .0153	.8500 ± .0513	.9463 ± .0153	.9350 ± .0513
vowel0	.9861 ± .0116	.9667 ± .0154	.9527 ± .0116	.9532 ± .0154	.9778 ± .0075	.9444 ± .0093	.9510 ± .0075	.9365 ± .0093	.9611 ± .0160	.9333 ± .0232	.9630 ± .0160	.9588 ± .0232
glass016vs2	.9560 ± .0558	.5500 ± .1228	.6443 ± .0558	.5886 ± .1228	.8231 ± .0355	.3000 ± .2385	.8257 ± .0355	.7886 ± .2385	.9714 ± .0171	.5833 ± .1741	.7286 ± .0171	.7086 ± .1741
glass2	.9264 ± .0256	.7667 ± .0841	.5330 ± .0256	.5074 ± .0841	.7626 ± .0615	.3500 ± .3725	.8514 ± .0615	.8027 ± .3725	.9407 ± .0533	.4667 ± .1544	.8225 ± .0533	.7505 ± .1544
ecoli0147vs2356	.9228 ± .0241	.7200 ± .1060	.9267 ± .0241	.8990 ± .1060	.8808 ± .0521	.7133 ± .1467	.9218 ± .0521	.9056 ± .1467	.9221 ± .0232	.8333 ± .0477	.9120 ± .0232	.9154 ± .0477
led7digit02456789vs1	.8582 ± .0246	.8607 ± .0829	.9421 ± .0246	.9483 ± .0829	.8582 ± .0274	.8321 ± .0708	.9434 ± .0274	.9507 ± .0708	.8648 ± .0216	.8607 ± .0851	.9403 ± .0216	.9458 ± .0851
glass06vs5	1.0000 ± .0073	1.0000 ± .0113	.9798 ± .0073	.9900 ± .0113	1.0000 ± .0069	.9000 ± .1332	.9849 ± .0069	.9300 ± .1332	1.0000 ± .0141	1.0000 ± .0215	.9722 ± .0141	.9595 ± .0215
ecoli01vs5	1.0000 ± .0137	.8000 ± .1248	.9636 ± .0137	.9682 ± .1248	1.0000 ± .0075	.9000 ± .0908	.9784 ± .0075	.9409 ± .0908	1.0000 ± .0043	.8500 ± .0868	.9784 ± .0043	.9545 ± .0868
glass0146vs2	.9253 ± .0483	.7833 ± .0450	.6208 ± .0483	.5909 ± .0450	.8978 ± .0665	.5167 ± .3191	.8085 ± .0665	.8027 ± .3191	.8956 ± .0273	.5833 ± .0748	.7486 ± .0273	.7343 ± .0748
ecoli0147vs56	.9700 ± .0222	.8000 ± .0385	.9455 ± .0222	.8987 ± .0385	.9900 ± .0154	.8000 ± .0661	.9577 ± .0154	.9282 ± .0661	.9800 ± .0198	.8400 ± .0898	.9381 ± .0198	.9118 ± .0898
cleveland0vs4	.9800 ± .0194	.5667 ± .1710	.9687 ± .0194	.9697 ± .1710	.9418 ± .0266	.6333 ± .2114	.9781 ± .0266	.9634 ± .2114	.9600 ± .0378	.8000 ± .1632	.9439 ± .0378	.9146 ± .1632
ecoli0146vs5	1.0000 ± .0171	.8500 ± .1133	.9577 ± .0171	.9385 ± .1133	.9875 ± .0168	.8500 ± .1158	.9750 ± .0168	.9423 ± .1158	1.0000 ± .0111	.8500 ± .1162	.9712 ± .0111	.9462 ± .1162
ecoli4	.9750 ± .0151	.9000 ± .0717	.9755 ± .0151	.9811 ± .0717	.9625 ± .0201	.8500 ± .0806	.9723 ± .0201	.9588 ± .0806	.9750 ± .0143	.9000 ± .0724	.9723 ± .0143	.9684 ± .0724
yeast1vs7	.9333 ± .0568	.8333 ± .0539	.5640 ± .0568	.5644 ± .0539	.8667 ± .0832	.4667 ± .1465	.8736 ± .0832	.8483 ± .1465	.9000 ± .0314	.6667 ± .0899	.7506 ± .0314	.6945 ± .0899
shuttle0vs4	1.0000 ± .0000	1.0000 ± .0020	1.0000 ± .0000	.9982 ± .0020	1.0000 ± .0003	.9920 ± .0103	.9990 ± .0003	.9988 ± .0103	.9980 ± .0023	.9917 ± .0094	1.0000 ± .0023	1.0000 ± .0094
glass4	1.0000 ± .0168	.7333 ± .4090	.9615 ± .0168	.9200 ± .4090	.9800 ± .0212	.8333 ± .1306	.9739 ± .0212	.9500 ± .1306	1.0000 ± .0187	.6667 ± .3937	.9478 ± .0187	.9200 ± .3937
page-blocks13vs4	.9273 ± .0477	.9000 ± .0679	.9189 ± .0477	.9144 ± .0679	.9391 ± .0703	.8400 ± .1574	.9262 ± .0703	.9233 ± .1574	.9735 ± .0116	.7933 ± .1273	.9825 ± .0116	.9752 ± .1273
abalone9-18	.7439 ± .0355	.7306 ± .0996	.6589 ± .0355	.6705 ± .0996	.8275 ± .0149	.5889 ± .1103	.7885 ± .0149	.7851 ± .1103	.8446 ± .0191	.7778 ± .0917	.8004 ± .0191	.7823 ± .0917
glass016vs5	1.0000 ± .0106	1.0000 ± .0422	.9643 ± .0106	.9314 ± .0422	1.0000 ± .0058	.9000 ± .1312	.9643 ± .0058	.9314 ± .1312	1.0000 ± .0108	.8000 ± .1672	.9557 ± .0108	.9429 ± .1672
shuttle2vs4	1.0000 ± .0496	1.0000 ± .0667	.9310 ± .0496	.9190 ± .0667	1.0000 ± .0046	1.0000 ± .0090	.9959 ± .0046	.9920 ± .0090	1.0000 ± .0046	1.0000 ± .0000	.9939 ± .0046	1.0000 ± .0000
yeast1458vs7	.6917 ± .1104	.4333 ± .2086	.6012 ± .1104	.5829 ± .2086	.7750 ± .0498	.4000 ± .1118	.7492 ± .0498	.7451 ± .1118	.8583 ± .0314	.5000 ± .1284	.6923 ± .0314	.6756 ± .1284
glass5	.9714 ± .0317	.6000 ± .5297	.9720 ± .0317	.9561 ± .5297	.9714 ± .0340	.6000 ± .4085	.9854 ± .0340	.9805 ± .4085	1.0000 ± .0368	.8000 ± .4225	.9183 ± .0368	.9122 ± .4225
yeast2vs8	.5750 ± .0319	.5500 ± .1487	.9919 ± .0319	.9957 ± .1487	.6500 ± .0211	.6000 ± .1606	.9973 ± .0211	.9978 ± .1606	.9625 ± .0175	.5500 ± .1322	.9259 ± .0175	.9111 ± .1322
yeast4	.8434 ± .0095	.7236 ± .0527	.8789 ± .0095	.8772 ± .0527	.7988 ± .0140	.6873 ± .0469	.8939 ± .0140	.8905 ± .0469	.9216 ± .0137	.8018 ± .0438	.8238 ± .0137	.8248 ± .0438
yeast1289vs7	.7583 ± .1556	.5333 ± .1253	.6065 ± .1556	.6122 ± .1253	.7917 ± .0672	.4667 ± .1633	.8277 ± .0672	.8079 ± .1633	.9000 ± .0495	.7000 ± .0902	.7132 ± .0495	.6925 ± .0902
yeast5	.9208 ± .0262	.8611 ± .0478	.9493 ± .0262	.9479 ± .0478	.9324 ± .0289	.8833 ± .0434	.9642 ± .0289	.9667 ± .0434	.9487 ± .0131	.9083 ± .0477	.9488 ± .0131	.9479 ± .0477
ecoli0137vs26	.8867 ± .0418	.7000 ± .4202	.9516 ± .0418	.9490 ± .4202	.8533 ± .0354	.7000 ± .4200	.9562 ± .0354	.9562 ± .4200	1.0000 ± .0106	.8000 ± .4201	.9362 ± .0106	.9015 ± .4201
yeast6	.8786 ± .0128	.8571 ± .0907	.9208 ± .0128	.9248 ± .0907	.8571 ± .0187	.8000 ± .1348	.9367 ± .0187	.9399 ± .1348	.8857 ± .0130	.8286 ± .0988	.9294 ± .0130	.9310 ± .0988
abalone19	.8508 ± .0131	.6952 ± .0824	.6165 ± .0131	.6200 ± .0824	.9298 ± .0196	.4714 ± .0569	.7402 ± .0196	.7359 ± .0569	.8588 ± .0165	.4667 ± .1476	.7233 ± .0165	.7216 ± .1476
Mean	.9097 ± .0307	**.7809 ± .1212**	.8643 ± .0307	.8531 ± .1212	.8983 ± .0267	.7319 ± .1334	.9231 ± .0267	**.9055 ± .1334**	.9398 ± .0204	.7797 ± .1233	.9025 ± .0204	.8855 ± .1233

Table 12
Complete table of results for GP-COACH versions with and without SMOTE preprocessing.

Data-set	No preprocessing						SMOTE preprocessing					
	GP-COACH-5		GP-COACH-9		GP-COACH-H		GP-COACH-5		GP-COACH-9		GP-COACH-H	
	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}
ecoli034vs5	.8348 ± .0623	.7293 ± .1508	.8436 ± .0337	.7854 ± .1903	.5871 ± .5366	.5412 ± .4967	.9755 ± .0165	.9018 ± .1282	.9761 ± .0158	.9250 ± .0709	.9833 ± .0276	.8660 ± .1252
yeast2vs4	.8111 ± .0405	.7934 ± .1018	.5085 ± .4646	.4557 ± .4232	.8855 ± .0243	.7817 ± .0850	.9252 ± .0092	.8987 ± .0412	.9283 ± .0044	.9036 ± .0381	.9647 ± .0095	.9304 ± .0288
ecoli067vs35	.7120 ± .1319	.5262 ± .3357	.8724 ± .0453	.6667 ± .3799	.9087 ± .0578	.6631 ± .3861	.9421 ± .0193	.8185 ± .2093	.9420 ± .0210	.8509 ± .2265	.9707 ± .0140	.7286 ± .4095
ecoli0234vs5	.7559 ± .2269	.6610 ± .3759	.8802 ± .0194	.7854 ± .1903	.2500 ± .4330	.1973 ± .4411	.9707 ± .0095	.8286 ± .1552	.9638 ± .0409	.8472 ± .1648	.9966 ± .0024	.8473 ± .1526
glass015vs2	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.5017 ± .1337	.3732 ± .2183	.8301 ± .0511	.2115 ± .3032	.9503 ± .0127	.6301 ± .0922
yeast0359vs78	.3816 ± .0722	.2151 ± .2033	.4467 ± .0198	.4595 ± .0941	.5025 ± .0540	.4136 ± .0989	.5764 ± .0840	.5111 ± .1244	.4804 ± .0258	.4467 ± .0833	.8919 ± .0188	.7189 ± .1013
yeast02579vs368	.4423 ± .2722	.4263 ± .3592	.2948 ± .0425	.2621 ± .1685	.9015 ± .0256	.8413 ± .0321	.9160 ± .0093	.9087 ± .0376	.9068 ± .0109	.9044 ± .0395	.9298 ± .0080	.9107 ± .0303
yeast0256vs3789	.3581 ± .0988	.1078 ± .1513	.3997 ± .1358	.2928 ± .2341	.7495 ± .0269	.6353 ± .1073	.8101 ± .0136	.7954 ± .0676	.8127 ± .0149	.7955 ± .0563	.8348 ± .0146	.7982 ± .0673
ecoli046vs5	.8263 ± .0684	.6878 ± .1833	.8725 ± .0144	.6514 ± .3834	.9551 ± .0180	.8105 ± .2129	.9744 ± .0168	.9171 ± .1248	.9855 ± .0174	.8793 ± .2166	.9952 ± .0039	.8677 ± .2102
ecoli01vs235	.7225 ± .1149	.5614 ± .3222	.7925 ± .1083	.5727 ± .3727	.4644 ± .2395	.2426 ± .3508	.9398 ± .0151	.8682 ± .1131	.9540 ± .0152	.9138 ± .0670	.9845 ± .0143	.8471 ± .0944
ecoli0267vs35	.8223 ± .0748	.5828 ± .3713	.8973 ± .0300	.8518 ± .1458	.7825 ± .3031	.6055 ± .3454	.9201 ± .0211	.8407 ± .1311	.9494 ± .0334	.8365 ± .1125	.9707 ± .0163	.9028 ± .0739
glass04vs5	.3917 ± .4067	.1414 ± .3162	.7777 ± .1085	.6243 ± .3713	.0000 ± .0000	.0000 ± .0000	.9662 ± .0208	.9064 ± .1277	.9706 ± .0247	.9632 ± .0134	.9909 ± .0125	.9429 ± .0419
ecoli0346vs5	.8211 ± .0317	.7560 ± .1884	.8353 ± .0390	.8005 ± .0897	.3742 ± .5123	.3732 ± .5132	.9959 ± .0028	.8772 ± .1132	.9842 ± .0159	.8934 ± .0632	.9993 ± .0015	.8847 ± .0690
ecoli0347vs56	.4067 ± .1337	.1779 ± .2436	.6756 ± .0664	.6729 ± .1658	.5673 ± .1876	.2654 ± .3956	.9576 ± .0104	.8525 ± .1039	.9644 ± .0166	.8571 ± .1459	.9881 ± .0036	.8767 ± .0977
yeast05679vs4	.5301 ± .1708	.4502 ± .2955	.0000 ± .0000	.0000 ± .0000	.7518 ± .1110	.5433 ± .1763	.8194 ± .0198	.8136 ± .0759	.8360 ± .0158	.8080 ± .0858	.8961 ± .0280	.6988 ± .0530
ecoli067vs5	.7960 ± .0691	.5861 ± .3606	.9076 ± .0156	.7474 ± .1815	.9218 ± .0186	.7505 ± .1554	.9545 ± .0144	.8356 ± .1277	.9554 ± .0162	.8923 ± .0884	.9849 ± .0041	.8671 ± .0629
vowel0	.7763 ± .0717	.7098 ± .0795	.8520 ± .0443	.8207 ± .0619	.8046 ± .1343	.7599 ± .1505	.9691 ± .0116	.9598 ± .0154	.9642 ± .0075	.9400 ± .0093	.9947 ± .0057	.9465 ± .0622
glass016vs2	.0555 ± .1240	.0000 ± .0000	.0555 ± .1240	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.7820 ± .0558	.5419 ± .1228	.8120 ± .0355	.4211 ± .2385	.9415 ± .0218	.6467 ± .2206
glass2	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0535 ± .1195	.0000 ± .0000	.7019 ± .0256	.6223 ± .0841	.7981 ± .0615	.3949 ± .3725	.9663 ± .0188	.5886 ± .1299
ecoli0147vs2356	.5083 ± .1123	.3942 ± .2360	.4271 ± .0894	.0816 ± .1826	.5234 ± .0427	.3036 ± .3019	.9247 ± .0241	.7976 ± .1060	.8994 ± .0521	.7972 ± .1467	.9594 ± .0153	.8263 ± .0687
led7digit02456789vs1	.8988 ± .0194	.9013 ± .0830	.9081 ± .0156	.8958 ± .0864	.5328 ± .4872	.5099 ± .4682	.8981 ± .0246	.9011 ± .0829	.8986 ± .0274	.8874 ± .0708	.9142 ± .0158	.9000 ± .0809
glass06vs5	.7986 ± .0637	.7548 ± .1379	.5142 ± .3470	.1949 ± .4359	.1512 ± .3381	.1414 ± .3162	.9898 ± .0073	.9949 ± .0113	.9924 ± .0069	.9060 ± .1332	.9975 ± .0035	.9120 ± .1263
ecoli01vs5	.7401 ± .0531	.6549 ± .1580	.7686 ± .1074	.4130 ± .3987	.9673 ± .0323	.8196 ± .1238	.9816 ± .0137	.8739 ± .1248	.9891 ± .0075	.9190 ± .0908	.9977 ± .0031	.8946 ± .0823
glass0146vs2	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.7552 ± .0483	.6651 ± .0450	.8501 ± .0665	.5675 ± .3191	.9313 ± .0074	.7300 ± .0534
ecoli0147vs56	.3948 ± .1736	.1789 ± .2449	.5243 ± .1376	.2444 ± .3541	.7058 ± .2287	.3563 ± .3715	.9574 ± .0222	.8472 ± .0385	.9735 ± .0154	.8577 ± .0661	.9852 ± .0045	.8372 ± .0595
cleveland0vs4	.8312 ± .0470	.5287 ± .3100	.8656 ± .0464	.3969 ± .5436	.5326 ± .3671	.2784 ± .3812	.9740 ± .0194	.7232 ± .1710	.9595 ± .0266	.7578 ± .2114	.9719 ± .0322	.8646 ± .1627
ecoli0146vs5	.6469 ± .0906	.4560 ± .4213	.8714 ± .0461	.6805 ± .2062	.9284 ± .0280	.7762 ± .2119	.9785 ± .0171	.8832 ± .1133	.9811 ± .0168	.8862 ± .1158	.9952 ± .0038	.9194 ± .0597
ecoli4	.6336 ± .1716	.5146 ± .3263	.6761 ± .3818	.4696 ± .3039	.0000 ± .0000	.0000 ± .0000	.9751 ± .0151	.9373 ± .0717	.9673 ± .0201	.9008 ± .0806	.9936 ± .0043	.9357 ± .0702
yeast1vs7	.0986 ± .1382	.0000 ± .0000	.2234 ± .2285	.0000 ± .0000	.1632 ± .1706	.0000 ± .0000	.7229 ± .0568	.6802 ± .0539	.8676 ± .0832	.6093 ± .1465	.8988 ± .0301	.6900 ± .0646
shuttle0vs4	.8361 ± .0143	.8300 ± .0595	.9877 ± .0101	.9744 ± .0387	1.0000 ± .0000	.9960 ± .0090	1.0000 ± .0000	.9991 ± .0020	.9995 ± .0003	.9954 ± .0103	1.0000 ± .0000	1.0000 ± .0000
glass4	.4689 ± .1175	.1155 ± .2582	.7019 ± .0955	.4737 ± .4550	.2963 ± .1948	.0000 ± .0000	.9804 ± .0168	.7231 ± .4090	.9766 ± .0212	.8811 ± .1306	.9906 ± .0077	.7303 ± .4132
page-blocks13vs4	.7063 ± .0430	.6917 ± .1876	.7397 ± .0877	.6815 ± .2216	.7321 ± .1134	.6016 ± .1727	.9205 ± .0477	.9035 ± .0679	.9316 ± .0703	.8706 ± .1574	.9994 ± .0008	.9482 ± .0502
abalone9-18	.3172 ± .0875	.2357 ± .2205	.3648 ± .2181	.2412 ± .2282	.4393 ± .1049	.2565 ± .2510	.6884 ± .0355	.6922 ± .0996	.8070 ± .0149	.6699 ± .1103	.8595 ± .0265	.7500 ± .0599
glass016vs5	.4309 ± .4010	.1414 ± .3162	.7755 ± .0843	.4828 ± .4567	.0000 ± .0000	.0000 ± .0000	.9819 ± .0106	.9644 ± .0422	.9820 ± .0058	.9090 ± .1312	.9921 ± .0078	.8550 ± .1596
shuttle2vs4	.6367 ± .2363	.6000 ± .5477	.2159 ± .3028	.2000 ± .4472	.8257 ± .1510	.8000 ± .4472	.9639 ± .0496	.9568 ± .0667	.9979 ± .0046	.9960 ± .0090	1.0000 ± .0000	.9918 ± .0183
yeast1458vs7	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.5775 ± .1104	.3546 ± .2086	.7566 ± .0498	.5353 ± .1118	.8952 ± .0261	.6304 ± .1095
glass5	.2926 ± .4010	.2000 ± .4472	.7766 ± .0977	.5372 ± .5051	.0000 ± .0000	.0000 ± .0000	.9711 ± .0317	.5801 ± .5297	.9780 ± .0340	.6758 ± .4085	.9957 ± .0027	.7877 ± .4404
yeast2vs8	.7401 ± .0348	.7283 ± .1497	.7401 ± .0348	.7283 ± .1497	.7410 ± .0351	.7283 ± .1497	.7544 ± .0319	.7274 ± .1487	.8049 ± .0211	.7601 ± .1606	.9937 ± .0047	.7381 ± .1765
yeast4	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0853 ± .1236	.0000 ± .0000	.8602 ± .0095	.7923 ± .0527	.8443 ± .0140	.7807 ± .0469	.9001 ± .0156	.8175 ± .0391
yeast1289vs7	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.6325 ± .1556	.5262 ± .1253	.7996 ± .0672	.5860 ± .1633	.8843 ± .0292	.6939 ± .1205
yeast5	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0333 ± .0745	.0000 ± .0000	.9344 ± .0262	.9020 ± .0478	.9477 ± .0289	.9229 ± .0434	.9724 ± .0066	.9428 ± .0526
ecoli0137vs26	.6472 ± .0986	.1414 ± .3162	.3344 ± .1877	.1414 ± .3162	.8430 ± .0583	.1401 ± .3133	.9167 ± .0418	.7215 ± .4202	.9021 ± .0354	.7203 ± .4200	.9843 ± .0107	.7067 ± .4136
yeast6	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.8994 ± .0128	.8856 ± .0907	.8960 ± .0187	.8574 ± .1348	.9319 ± .0155	.8170 ± .0977
abalone19	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.0000 ± .0000	.7196 ± .0131	.6425 ± .0824	.8290 ± .0196	.5828 ± .0569	.8558 ± .0193	.5532 ± .1487
Mean	.4789 ± .1017	.3677 ± .1922	.5074 ± .0871	.3929 ± .1996	.4536 ± .1216	.3439 ± .1697	.8763 ± .0307	.7897 ± .1212	.9056 ± .0267	.7845 ± .1334	.9576 ± .0121	**.8175 ± .1193**

Table 13
Complete table of results for FRBCS methods and C4.5 in highly imbalanced data-sets. SMOTE preprocessing for FRBCS methods, SMOTE + ENN for C4.5.

Data-set	GP-COACH-5		GP-COACH-9		HFRBCS(Chi)		GP-COACH-H		C4.5	
	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}
ecoli034vs5	.9755 ± .0165	.9018 ± .1282	.9761 ± .0158	.9250 ± .0709	.9930 ± .0050	.8421 ± .1471	.9833 ± .0276	.8660 ± .1252	.9762 ± .0149	.8761 ± .0492
yeast2vs4	.9252 ± .0092	.8987 ± .0412	.9283 ± .0044	.9036 ± .0381	.9527 ± .0105	.8932 ± .0418	.9647 ± .0095	.9304 ± .0288	.9745 ± .0066	.9029 ± .0406
ecoli067vs35	.9421 ± .0193	.8185 ± .2093	.9420 ± .0210	.8509 ± .2265	.9574 ± .0163	.8267 ± .1415	.9707 ± .0140	.7286 ± .4095	.9771 ± .0204	.7206 ± .4072
ecoli0234vs5	.9707 ± .0095	.8286 ± .1552	.9638 ± .0409	.8472 ± .1648	.9910 ± .0031	.8425 ± .1504	.9966 ± .0024	.8473 ± .1526	.9827 ± .0074	.8861 ± .1245
glass015vs2	.5017 ± .1337	.3732 ± .2183	.8301 ± .0511	.2115 ± .3032	.6967 ± .0269	.5590 ± .0851	.9503 ± .0127	.6301 ± .0922	.9066 ± .0263	.7788 ± .2089
yeast0359vs78	.5764 ± .0840	.5111 ± .1244	.4804 ± .0258	.4467 ± .0833	.8401 ± .0040	.7330 ± .0403	.8919 ± .0188	.7189 ± .1013	.9213 ± .0214	.6894 ± .0888
yeast02579vs368	.9160 ± .0093	.9087 ± .0376	.9068 ± .0109	.9044 ± .0395	.9063 ± .0103	.8946 ± .0355	.9298 ± .0080	.9107 ± .0303	.9572 ± .0206	.9125 ± .0336
yeast0256vs3789	.8101 ± .0136	.7954 ± .0676	.8127 ± .0149	.7955 ± .0563	.8106 ± .0165	.7927 ± .0674	.8348 ± .0146	.7982 ± .0673	.9173 ± .0180	.7707 ± .0366
ecoli046vs5	.9744 ± .0168	.9171 ± .1248	.9855 ± .0174	.8793 ± .2166	.9925 ± .0029	.8800 ± .1156	.9952 ± .0039	.8677 ± .2102	.9834 ± .0079	.8776 ± .1148
ecoli01vs235	.9398 ± .0151	.8682 ± .1131	.9540 ± .0152	.9138 ± .0670	.9844 ± .0132	.8709 ± .1076	.9845 ± .0143	.8471 ± .0944	.9649 ± .0302	.8277 ± .1191
ecoli0267vs35	.9201 ± .0211	.8407 ± .1311	.9494 ± .0334	.8365 ± .1125	.9609 ± .0140	.8247 ± .1089	.9707 ± .0163	.9028 ± .0739	.9825 ± .0058	.8061 ± .1065
glass04vs5	.9662 ± .0208	.9064 ± .1277	.9706 ± .0247	.9632 ± .0134	.9457 ± .0185	.7092 ± .3976	.9909 ± .0125	.9429 ± .0419	.9909 ± .0063	.9748 ± .0269
ecoli0346vs5	.9959 ± .0028	.8772 ± .1132	.9842 ± .0159	.8934 ± .0632	.9918 ± .0057	.8729 ± .1175	.9993 ± .0015	.8847 ± .0690	.9884 ± .0046	.8946 ± .0793
ecoli0347vs56	.9576 ± .0104	.8525 ± .1039	.9644 ± .0166	.8571 ± .1459	.9682 ± .0099	.9007 ± .0962	.9881 ± .0036	.8767 ± .0977	.9566 ± .0176	.8413 ± .1377
yeast05679vs4	.8194 ± .0198	.8136 ± .0759	.8360 ± .0158	.8080 ± .0858	.9290 ± .0103	.7318 ± .0747	.8961 ± .0280	.6988 ± .0530	.9197 ± .0128	.7678 ± .1029
ecoli067vs5	.9545 ± .0144	.8356 ± .1277	.9554 ± .0162	.8923 ± .0884	.9531 ± .0211	.8559 ± .0542	.9849 ± .0041	.8671 ± .0629	.9740 ± .0103	.8376 ± .1167
vowel0	.9691 ± .0116	.9598 ± .0154	.9642 ± .0075	.9400 ± .0093	.9999 ± .0003	.9882 ± .0162	.9947 ± .0057	.9465 ± .0622	.9943 ± .0047	.9417 ± .0815
glass016vs2	.7820 ± .0558	.5419 ± .1228	.8120 ± .0355	.4211 ± .2385	.8726 ± .0230	.5837 ± .2004	.9415 ± .0218	.6467 ± .2206	.9365 ± .0323	.6063 ± .1173
glass2	.7019 ± .0256	.6223 ± .0841	.7981 ± .0615	.3949 ± .3725	.8299 ± .0174	.5484 ± .2057	.9663 ± .0188	.5886 ± .1299	.9261 ± .0342	.7377 ± .1633
ecoli0147vs2356	.9247 ± .0241	.7976 ± .1060	.8994 ± .0521	.7972 ± .1467	.9517 ± .0109	.8477 ± .0655	.9594 ± .0153	.8263 ± .0687	.9563 ± .0318	.8119 ± .0459
led7digit02456789vs1	.8981 ± .0246	.9011 ± .0829	.8986 ± .0274	.8874 ± .0708	.9380 ± .0212	.8276 ± .0778	.9142 ± .0158	.9000 ± .0809	.9217 ± .0192	.8370 ± .0475
glass06vs5	.9898 ± .0073	.9949 ± .0113	.9924 ± .0069	.9060 ± .1332	.9744 ± .0046	.8907 ± .1178	.9975 ± .0035	.9120 ± .1263	.9911 ± .0035	.9628 ± .0556
ecoli01vs5	.9816 ± .0137	.8739 ± .1248	.9891 ± .0075	.9190 ± .0908	.9932 ± .0043	.8689 ± .1166	.9977 ± .0031	.8946 ± .0823	.9828 ± .0068	.8081 ± .1213
glass0146vs2	.7552 ± .0483	.6651 ± .0450	.8501 ± .0665	.5675 ± .3191	.7005 ± .0077	.5117 ± .1026	.9313 ± .0074	.7300 ± .0534	.9010 ± .0596	.6157 ± .3465
ecoli0147vs56	.9574 ± .0222	.8472 ± .0385	.9735 ± .0154	.8577 ± .0661	.9790 ± .0059	.8886 ± .0918	.9852 ± .0045	.8372 ± .0595	.9608 ± .0173	.8250 ± .1380
cleveland0vs4	.9740 ± .0194	.7232 ± .1710	.9595 ± .0266	.7578 ± .2114	.9992 ± .0018	.3961 ± .3827	.9719 ± .0322	.8646 ± .1627	.9819 ± .0187	.7307 ± .1517
ecoli0146vs5	.9785 ± .0171	.8832 ± .1133	.9811 ± .0168	.8862 ± .1158	.9913 ± .0047	.8674 ± .1069	.9952 ± .0038	.9194 ± .0597	.9850 ± .0061	.8880 ± .1148
ecoli4	.9751 ± .0151	.9373 ± .0717	.9673 ± .0201	.9008 ± .0806	.9869 ± .0141	.9302 ± .0817	.9936 ± .0043	.9357 ± .0702	.9826 ± .0170	.8947 ± .1202
yeast1vs7	.7229 ± .0568	.6802 ± .0539	.8676 ± .0832	.6093 ± .1465	.9163 ± .0225	.7074 ± .1240	.8988 ± .0301	.6900 ± .0646	.9093 ± .0332	.7222 ± .0532
shuttle0vs4	1.0000 ± .0000	.9991 ± .0020	.9995 ± .0003	.9954 ± .0103	1.0000 ± .0000	.9912 ± .0115	1.0000 ± .0000	1.0000 ± .0000	.9999 ± .0002	.9997 ± .0007
glass4	.9804 ± .0168	.7231 ± .4090	.9766 ± .0212	.8811 ± .1306	.9981 ± .0017	.7039 ± .4049	.9906 ± .0077	.7303 ± .4132	.9665 ± .0149	.7639 ± .4279
page-blocks13vs4	.9205 ± .0477	.9035 ± .0679	.9316 ± .0703	.8706 ± .1574	.9989 ± .0012	.9864 ± .0065	.9994 ± .0008	.9482 ± .0502	.9975 ± .0018	.9909 ± .0065
abalone9-18	.6884 ± .0355	.6922 ± .0996	.8070 ± .0149	.6699 ± .1103	.8396 ± .0303	.6756 ± .1401	.8595 ± .0265	.7500 ± .0599	.9273 ± .0074	.6884 ± .1181
glass016vs5	.9819 ± .0106	.9644 ± .0422	.9820 ± .0058	.9090 ± .1312	.9971 ± .0030	.7796 ± .4361	.9921 ± .0078	.8550 ± .1596	.9863 ± .0047	.7738 ± .4328
shuttle2vs4	.9639 ± .0496	.9568 ± .0667	.9979 ± .0046	.9960 ± .0090	.9990 ± .0023	.9749 ± .0271	1.0000 ± .0000	.9918 ± .0183	1.0000 ± .0000	1.0000 ± .0000
yeast1458vs7	.5775 ± .1104	.3546 ± .2086	.7566 ± .0498	.5353 ± .1118	.9037 ± .0133	.6249 ± .0626	.8952 ± .0261	.6304 ± .1095	.8717 ± .0492	.3345 ± .3342
glass5	.9711 ± .0317	.5801 ± .5297	.9780 ± .0340	.6758 ± .4085	.9764 ± .0221	.6873 ± .3956	.9957 ± .0027	.7877 ± .4404	.9698 ± .0296	.5851 ± .5343
yeast2vs8	.7544 ± .0319	.7274 ± .1487	.8049 ± .0211	.7601 ± .1606	.8334 ± .0164	.7247 ± .1510	.9937 ± .0047	.7381 ± .1765	.8923 ± .0447	.8033 ± .1167
yeast4	.8602 ± .0095	.7923 ± .0527	.8443 ± .0140	.7807 ± .0469	.9001 ± .0194	.8264 ± .0229	.9001 ± .0156	.8175 ± .0391	.8984 ± .0123	.6897 ± .0769
yeast1289vs7	.6325 ± .1556	.5262 ± .1253	.7996 ± .0672	.5860 ± .1633	.8699 ± .0224	.6937 ± .0437	.8843 ± .0292	.6939 ± .1205	.9408 ± .0259	.5522 ± .1662
yeast5	.9344 ± .0262	.9020 ± .0478	.9477 ± .0289	.9229 ± .0434	.9782 ± .0033	.9420 ± .0259	.9724 ± .0066	.9428 ± .0526	.9819 ± .0077	.9390 ± .0474
ecoli0137vs26	.9167 ± .0418	.7215 ± .4202	.9021 ± .0354	.7203 ± .4200	.9867 ± .0079	.7148 ± .4180	.9843 ± .0107	.7067 ± .4136	.9650 ± .0320	.7062 ± .4093
yeast6	.8994 ± .0128	.8856 ± .0907	.8960 ± .0187	.8574 ± .1348	.9341 ± .0177	.8492 ± .1288	.9319 ± .0155	.8170 ± .0977	.9301 ± .0157	.8029 ± .1541
abalone19	.7196 ± .0131	.6425 ± .0824	.8290 ± .0196	.5828 ± .0569	.8343 ± .0280	.7019 ± .0856	.8558 ± .0193	.5532 ± .1487	.8838 ± .0300	.1550 ± .2125
Mean	.8763 ± .0307	.7897 ± .1212	.9056 ± .0267	.7845 ± .1334	.9331 ± .0117	.7901 ± .1325	.9576 ± .0121	**.8175 ± .1193**	.9549 ± .0180	.7848 ± .1452

Table 14
Complete table of results for FRBCS methods and C4.5 in borderline imbalanced data-sets. SMOTE preprocessing for FRBCS methods, SMOTE+ENN for C4.5.

Data-set	GP-COACH-5		GP-COACH-9		HFRBCS(Chi)		GP-COACH-H		C4.5	
	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}	GM_{tr}	GM_{tst}
paw02a-600-5-70-BI	.7969 ± .0223	.7892 ± .0297	.7898 ± .0311	.7178 ± .0867	.8730 ± .0129	.8460 ± .0395	.8824 ± .0068	.8523 ± .0342	.8879 ± .0176	.8310 ± .0292
paw02a-600-5-60-BI	.7849 ± .0185	.7603 ± .0698	.7364 ± .1198	.6329 ± .1650	.8485 ± .0311	.8157 ± .0309	.8753 ± .0046	.8501 ± .0235	.8755 ± .0287	.8094 ± .0279
paw02a-600-5-50-BI	.8188 ± .0129	.7869 ± .0676	.8541 ± .0123	.8117 ± .0348	.8595 ± .0307	.8226 ± .0514	.9002 ± .0067	.8402 ± .0227	.8936 ± .0213	.8301 ± .0468
paw02a-600-5-30-BI	.8418 ± .0121	.8193 ± .0639	.8567 ± .0209	.8281 ± .0695	.8749 ± .0185	.8573 ± .0352	.9036 ± .0143	.8605 ± .0660	.8990 ± .0328	.8604 ± .0542
paw02a-600-5-0-BI	.8427 ± .0313	.8489 ± .0311	.9130 ± .0299	.8765 ± .0836	.9339 ± .0076	.9142 ± .0179	.9596 ± .0055	.9367 ± .0124	.9512 ± .0389	.9473 ± .0259
04clover5z-600-5-70-BI	.7706 ± .0217	.7457 ± .1094	.7832 ± .0233	.7094 ± .0921	.7790 ± .0371	.7427 ± .0816	.8250 ± .0126	.7795 ± .0574	.8665 ± .0173	.7557 ± .0468
04clover5z-600-5-60-BI	.7789 ± .0209	.7464 ± .0290	.7243 ± .0473	.6445 ± .0800	.7713 ± .0242	.7725 ± .0514	.8405 ± .0146	.7986 ± .0328	.8895 ± .0151	.7990 ± .0757
04clover5z-600-5-50-BI	.7658 ± .0362	.7434 ± .0872	.7804 ± .0385	.7496 ± .1121	.7925 ± .0390	.7582 ± .0536	.8537 ± .0280	.8080 ± .0530	.8771 ± .0286	.8063 ± .0575
04clover5z-600-5-30-BI	.7839 ± .0294	.7683 ± .0638	.8029 ± .0699	.7453 ± .0767	.8087 ± .0332	.8097 ± .0474	.8737 ± .0182	.8093 ± .0688	.8957 ± .0175	.8291 ± .0337
04clover5z-600-5-0-BI	.7842 ± .0151	.7663 ± .0536	.8389 ± .0146	.7705 ± .0179	.8104 ± .0288	.8008 ± .0309	.8900 ± .0363	.8519 ± .0517	.9269 ± .0333	.8652 ± .0271
03subcl5-600-5-70-BI	.6807 ± .0416	.6319 ± .0582	.7940 ± .0278	.7478 ± .0572	.7947 ± .0138	.7278 ± .0277	.8528 ± .0098	.8006 ± .0419	.8381 ± .0367	.7617 ± .0406
03subcl5-600-5-60-BI	.7471 ± .0251	.6898 ± .0903	.8047 ± .0196	.7789 ± .0483	.8083 ± .0182	.7498 ± .0719	.8067 ± .0434	.7379 ± .0545	.8322 ± .0348	.7641 ± .0732
03subcl5-600-5-50-BI	.7688 ± .0094	.7320 ± .0902	.7920 ± .0172	.7500 ± .0869	.8020 ± .0227	.7465 ± .0537	.8269 ± .0257	.7563 ± .0600	.8332 ± .0109	.7753 ± .0895
03subcl5-600-5-30-BI	.8022 ± .0333	.7860 ± .0504	.8183 ± .0137	.8020 ± .0930	.8148 ± .0276	.7713 ± .0552	.8474 ± .0169	.8307 ± .0379	.8615 ± .0208	.8140 ± .0552
03subcl5-600-5-0-BI	.8795 ± .0173	.8685 ± .0470	.8952 ± .0145	.8965 ± .0183	.8985 ± .0048	.8765 ± .0329	.9364 ± .0015	.9179 ± .0132	.9336 ± .0265	.8969 ± .0368
paw02a-800-7-70-BI	.7947 ± .0138	.7733 ± .0423	.7872 ± .0391	.6775 ± .1242	.8601 ± .0118	.8197 ± .0415	.8741 ± .0104	.8421 ± .0274	.8923 ± .0147	.8001 ± .0486
paw02a-800-7-60-BI	.8028 ± .0132	.7410 ± .0558	.8089 ± .0167	.7235 ± .0549	.8514 ± .0070	.8113 ± .0439	.8706 ± .0105	.8253 ± .0523	.8817 ± .0188	.8043 ± .0352
paw02a-800-7-50-BI	.8211 ± .0026	.7736 ± .0546	.8317 ± .0152	.7905 ± .0352	.8719 ± .0074	.8373 ± .0416	.8863 ± .0081	.8164 ± .0704	.9072 ± .0238	.8448 ± .0447
paw02a-800-7-30-BI	.8391 ± .0135	.8170 ± .0565	.8445 ± .0423	.7998 ± .0312	.8894 ± .0059	.8672 ± .0145	.9030 ± .0077	.8299 ± .0398	.9135 ± .0282	.8449 ± .0455
paw02a-800-7-0-BI	.8493 ± .0472	.8300 ± .0418	.9197 ± .0506	.9100 ± .0433	.9288 ± .0068	.9307 ± .0192	.9576 ± .0092	.9351 ± .0245	.9532 ± .0318	.9371 ± .0334
04clover5z-800-7-70-BI	.7708 ± .0199	.7351 ± .0673	.7839 ± .0335	.7100 ± .1069	.7898 ± .0143	.7237 ± .1008	.8182 ± .0161	.7857 ± .0643	.8723 ± .0429	.7525 ± .0956
04clover5z-800-7-60-BI	.7722 ± .0159	.7796 ± .0340	.7108 ± .0542	.6688 ± .0708	.7799 ± .0194	.7842 ± .0591	.8256 ± .0073	.7707 ± .0461	.8889 ± .0276	.7704 ± .0396
04clover5z-800-7-50-BI	.7860 ± .0215	.7436 ± .0766	.7744 ± .0198	.7544 ± .1039	.7928 ± .0290	.7543 ± .0608	.8390 ± .0138	.7897 ± .0645	.8946 ± .0099	.8261 ± .0701
04clover5z-800-7-30-BI	.7879 ± .0219	.7488 ± .0362	.8162 ± .0221	.7545 ± .0775	.8075 ± .0324	.7750 ± .0583	.8513 ± .0171	.7805 ± .0541	.8930 ± .0215	.8256 ± .0360
04clover5z-800-7-0-BI	.7962 ± .0210	.7578 ± .0503	.8249 ± .0347	.7694 ± .0373	.8091 ± .0283	.7693 ± .0608	.8958 ± .0213	.8541 ± .0578	.9412 ± .0370	.8730 ± .0413
03subcl5-800-7-70-BI	.6662 ± .0352	.6456 ± .0446	.7107 ± .1049	.6454 ± .1444	.7784 ± .0059	.7552 ± .0490	.8186 ± .0330	.7868 ± .0250	.8255 ± .0211	.7735 ± .0376
03subcl5-800-7-60-BI	.7250 ± .0225	.6991 ± .0392	.7962 ± .0137	.7696 ± .0284	.7896 ± .0195	.7331 ± .0436	.8102 ± .0325	.7729 ± .0347	.8374 ± .0244	.7513 ± .0357
03subcl5-800-7-50-BI	.7584 ± .0152	.7261 ± .0908	.8021 ± .0058	.7516 ± .0509	.7927 ± .0158	.7239 ± .0340	.8204 ± .0200	.7436 ± .0310	.8396 ± .0089	.7507 ± .0438
03subcl5-800-7-30-BI	.7993 ± .0219	.7689 ± .0630	.8133 ± .0269	.7901 ± .0594	.8327 ± .0228	.7955 ± .0439	.8552 ± .0081	.8297 ± .0390	.8812 ± .0155	.7941 ± .0220
03subcl5-800-7-0-BI	.8814 ± .0223	.8663 ± .0396	.8998 ± .0099	.9061 ± .0238	.9036 ± .0081	.8851 ± .0321	.9217 ± .0126	.9081 ± .0224	.9588 ± .0249	.9292 ± .0370
Mean	.7899 ± .0218	.7630 ± .0578	.8103 ± .0330	.7628 ± .0705	.8316 ± .0195	.7992 ± .0461	.8674 ± .0157	**.8234** ± **.0428**	.8881 ± .0244	.8208 ± .0462

References

[1] R. Alcalá, J. Alcalá-Fdez, F. Herrera, A proposal for the genetic lateral tuning of linguistic fuzzy systems and its interaction with rule selection, IEEE Transactions on Fuzzy Systems 15 (2007) 616–635.

[2] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, Journal of Multi-Valued Logic and Soft Computing 17 (2011) 255–287.

[3] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Computing 13 (2009) 307–318.

[4] R. Barandela, J.S. Sánchez, V. García, E. Rangel, Strategies for learning in class imbalance problems, Pattern Recognition 36 (2003) 849–851.

[5] A. Bastian, How to handle the flexibility of linguistic variables with applications, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 2 (1994) 463–484.

[6] G.E.A.P.A. Batista, R.C. Prati, M.C. Monard, A study of the behaviour of several methods for balancing machine learning training data, SIGKDD Explorations 6 (2004) 20–29.

[7] F.J. Berlanga, A.J. Rivera, M.J. del Jesus, F. Herrera, GP-COACH: genetic programming-based learning of compact and accurate fuzzy rule-based classification systems for high-dimensional problems, Information Sciences 180 (2010) 1183–1200.

[8] J. Charles, Automatic recognition of complete palynomorphs in digital images, Machine Vision and Applications 22 (2011) 53–60.

[9] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, Journal of Artificial Intelligent Research 16 (2002) 321–357.

[10] Z. Chi, H. Yan, T. Pham, Fuzzy Algorithms with Applications to Image Processing and Pattern Recognition, World Scientific, 1996.

[11] O. Cordón, F. Herrera, A proposal for improving the accuracy of linguistic modeling, IEEE Transactions on Fuzzy Systems 8 (2000) 335–344.

[12] O. Cordón, F. Herrera, F. Hoffmann, L. Magdalena, Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases, Advances in Fuzzy Systems – Applications and Theory, vol. 19, World Scientific, 2001.

[13] O. Cordón, F. Herrera, P. Villar, Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base, IEEE Transactions on Fuzzy Systems 9 (2001) 667–674.

[14] O. Cordon, F. Herrera, I. Zwir, Linguistic modeling by hierarchical systems of linguistic rules, IEEE Transactions on Fuzzy Systems 10 (2002) 2–20.

[15] O. Cordón, M. del Jesus, F. Herrera, A proposal on reasoning methods in fuzzy rule-based classification systems, International Journal of Approximate 20 (1999) 21–45.

[16] C. Drummond, R.C. Holte, C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling, in: Proceedings of the International Conference on Machine Learning (ICML 2003) Workshop on Learning from Imbalanced Data Sets II, 2003, pp. 1–8.

[17] C. Elkan, The foundations of cost-sensitive learning, in: Proceedings of the 17th IEEE International Joint Conference on Artificial Intelligence (IJCAI'01), 2001, pp. 973–978.

[18] L.J. Eshelman, The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination, in: Foudations of Genetic Algorithms, Morgan Kaufmann, 1990, pp. 265–283.

[19] A. Fernández, S. García, F. Herrera, Addressing the classification with imbalanced data: open problems and new challenges on class distribution, in: Proceedings of the 6th International Conference on Hybrid Artificial Intelligence Systems (HAIS'11), 2011, pp. 1–10.

[20] A. Fernández, S. García, M.J. del Jesus, F. Herrera, A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets, Fuzzy Sets and Systems 159 (2008) 2378–2398.

[21] A. Fernández, M.J. del Jesus, F. Herrera, Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets, International Journal of Approximate Reasoning 50 (2009) 561–577.

[22] A. Fernández, M.J. del Jesus, F. Herrera, On the 2-tuples based genetic tuning performance for fuzzy rule based classification systems in imbalanced data-sets, Information Sciences 180 (2010) 1268–1291.

[23] S. García, J. Derrac, I. Triguero, C.J. Carmona, F. Herrera, Evolutionary-based selection of generalized instances for imbalanced classification, Knowledge-Based Systems 25 (2012) 3–12.

[24] S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Computing 13 (2009) 959–977.

[25] S. García, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, Journal of Machine Learning Research 9 (2008) 2607–2624.

[26] V. García, R. Mollineda, J.S. Sánchez, On the k-NN performance in a challenging scenario of imbalance and overlapping, Pattern Analysis Applications 11 (2008) 269–280.

[27] V. García, J. Sánchez, R. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Knowledge-Based Systems 25 (2012) 13–21.

[28] N. García-Pedrajas, J. Pérez-Rodríguez, M. García-Pedrajas, D. Ortiz-Boyer, C. Fyfe, Class imbalance methods for translation initiation site recognition in DNA sequences, Knowledge-Based Systems 25 (2012) 22–34.

[29] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Transactions on Knowledge and Data Engineering 21 (2009) 1263–1284.

[30] F. Herrera, Genetic fuzzy systems: taxonomy, current research trends and prospects, Evolutionary Intelligence 1 (2008) 27–46.

[31] F. Herrera, M. Lozano, A.M. Sánchez, A taxonomy for the crossover operator for real-coded genetic algorithms: an experimental study, International Journal of Intelligent Systems 18 (2003) 309–338.

[32] F. Herrera, L. Martínez, A 2-tuple fuzzy linguistic representation model for computing with words, IEEE Transactions on Fuzzy Systems 8 (2000) 746–752.

[33] H. Ishibuchi, T. Nakashima, Effect of rule weights in fuzzy rule-based classification systems, IEEE Transactions on Fuzzy Systems 9 (2001) 506–515.

[34] H. Ishibuchi, T. Nakashima, M. Nii, Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining, Springer-Verlag, 2004.

[35] H. Ishibuchi, K. Nozaki, N. Yamamoto, H. Tanaka, Selecting fuzzy if–then rules from classification problems using genetic algorithms, IEEE Transactions on Fuzzy Systems 9 (1995) 260–270.

[36] H. Ishibuchi, T. Yamamoto, Rule weight specification in fuzzy rule-based classification systems, IEEE Transactions on Fuzzy Systems 13 (2005) 428–435.

[37] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intelligent Data Analysis Journal 6 (2002) 429–450.

[38] W. Khreich, E. Granger, A. Miri, R. Sabourin, Iterative boolean combination of classifiers in the ROC space: an application to anomaly detection with HMMs, Pattern Recognition 43 (2010) 2732–2752.

[39] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, The MIT Press, 1992.

[40] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: one-sided selection, in: 14th International Conference on Machine Learning(ICML97), 1997, pp. 179–186.

[41] J. Liu, Q. Hu, D. Yu, A comparative study on rough set based class imbalance learning, Knowledge-Based Systems 21 (2008) 753–763.

[42] V. López, A. Fernández, J.G. Moreno-Torres, F. Herrera, Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics, Expert Systems with Applications 39 (2012) 6585–6608.

[43] L. Magdalena, F. Monasterio-Huelin, A fuzzy logic controller with learning through the evolution of its knowledge base, International Journal of Approximate Reasoning 16 (1997) 335–358.

[44] K. Napierala, J. Stefanowski, S. Wilk, Learning from imbalanced data in presence of noisy and borderline examples, in: 7th International Conference on Rough Sets and Current Trends in Computing (RSCTC2010), 2010, pp. 158–167.

[45] S. Oh, M. Lee, B.T. Zhang, Ensemble learning with active example selection for imbalanced biomedical data classification, IEEE/ACM Transactions on Computational Biology and Bioinformatics 8 (2011) 316–325.

[46] A. Orriols-Puig, E. Bernadó-Mansilla, Evolutionary rule-based systems for imbalanced datasets, Soft Computing 13 (2009) 213–225.

[47] J. Quinlan, C4.5: Programs for Machine Learning., Morgan Kauffman, 1993.

[48] J. Ren, ANN vs. SVM: which one performs better in classification of MCCs in mammogram imaging, Knowledge-Based Systems 26 (2012) 144–153.

[49] C. Seiffert, T.M. Khoshgoftaar, J.V. Hulse, A. Folleco, An empirical study of the classification performance of learners on imbalanced and noisy software quality data, Information Sciences, in press. http://dx.doi.org/10.1016/j.ins.2010.12.016.

[50] D. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman & Hall/CRC, 2011.

[51] Y. Sun, A.K.C. Wong, M.S. Kamel, Classification of imbalanced data: a review, International Journal of Pattern Recognition and Artificial Intelligence 23 (2009) 687–719.

[52] D. Wang, T.S. Dillon, Extraction of classification rules characterized by ellipsoidal regions using soft-computing techniques, International Journal of Systems Science 37 (2006) 969–980.

[53] G.M. Weiss, Mining with rarity: a unifying framework, SIGKDD Explorations 6 (2004) 7–19.

[54] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man and Cybernetics 2 (1972) 408–421.

[55] M.L. Wong, K.S. Leung, Data Mining Using Grammar-Based Genetic Programming and Applications, Kluwer Academic Publishers, 2000.

[56] Q. Yang, X. Wu, 10 challenging problems in data mining research, International Journal of Information Technology & Decision Making 5 (2006) 597–604.

[57] B. Zadrozny, C. Elkan, Learning and making decisions when costs and probabilities are both unknown, in: Proceedings of the 7th International Conference on Knowledge Discovery and Data Mining (KDD'01), 2001, pp. 204–213.

2.2. On the Importance of the Validation Technique for Classification with Imbalanced Datasets: Addressing Covariate Shift when Data is Skewed

- V. López, A. Fernández, F. Herrera, On the Importance of the Validation Technique for Classification with Imbalanced Datasets: Addressing Covariate Shift when Data is Skewed. Information Sciences 257 (2014) 1–13, doi: 10.1016/j.ins.2013.09.038

  - Status: **Published**.
  - Impact Factor (JCR 2012): 3.643.
  - Subject Category: Computer Science, Information Systems. Ranking 6 / 132 (**Q1**).

Information Sciences 257 (2014) 1–13

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

On the importance of the validation technique for classification with imbalanced datasets: Addressing covariate shift when data is skewed

Victoria López [a,*], Alberto Fernández [b], Francisco Herrera [a]

[a] Department of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, Granada, Spain
[b] Department of Computer Science, University of Jaén, Jaén, Spain

ARTICLE INFO

Article history:
Received 13 August 2012
Received in revised form 14 June 2013
Accepted 15 September 2013
Available online 21 September 2013

Keywords:
Classification
Imbalanced dataset
Covariate shift
Dataset shift
Validation technique
Partitioning

ABSTRACT

In the field of Data Mining, the estimation of the quality of the learned models is a key step in order to select the most appropriate tool for the problem to be solved. Traditionally, a *k*-fold validation technique has been carried out so that there is a certain degree of independency among the results for the different partitions. In this way, the highest average performance will be obtained by the most robust approach. However, applying a "random" division of the instances over the folds may result in a problem known as dataset shift, which consists in having a different data distribution between the training and test folds.

In classification with imbalanced datasets, in which the number of instances of one class is much lower than the other class, this problem is more severe. The misclassification of minority class instances due to an incorrect learning of the real boundaries caused by a not well fitted data distribution, truly affects the measures of performance in this scenario. Regarding this fact, we propose the use of a specific validation technique for the partitioning of the data, known as "Distribution optimally balanced stratified cross-validation" to avoid this harmful situation in the presence of imbalance. This methodology makes the decision of placing close-by samples on different folds, so that each partition will end up with enough representatives of every region.

We have selected a wide number of imbalanced datasets from KEEL dataset repository for our study, using several learning techniques from different paradigms, thus making the conclusions extracted to be independent of the underlying classifier. The analysis of the results has been carried out by means of the proper statistical study, which shows the goodness of this approach for dealing with imbalanced data.

1. Introduction

Standard learning algorithms are designed under the premise of a balanced class distribution. When dealing with skewed class distributions, the classification problem becomes more difficult, specifically for correctly identifying the minority concepts within the data [11]. This issue is known as the class imbalance problem [21,38], in which there is an under-represented class (positive) and a majority class (negative). This problem is present in many real-world classification tasks and has been considered as a challenge within the Data Mining community [48].

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: vlopez@decsai.ugr.es (V. López), alberto.fernandez@ujaen.es (A. Fernández), herrera@decsai.ugr.es (F. Herrera).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.09.038

In order to validate the performance of a classifier, both in standard and imbalanced classification, stratified cross-validation (SCV) is the most commonly employed method in the literature. It places an equal number of samples of each class on each partition to maintain class distributions similar in all partitions [9]. However, when this process is carried out in a random way, it may introduce a different data distribution between the training and test partitions, thus leading to inaccurate conclusions when learning a model from the training data. This issue is known as dataset shift [8], or more specifically covariate shift [30].

In the presence of imbalance, this problem is even more critic according to the metrics of performance applied in this scenario. Since misclassifications for the positive class instances severely hinder the average precision, we must try to avoid those errors in test which are due to a "random clustering" of the classes, i.e. generating outliers.

A more suitable validation technique needs to be employed in order to avoid introducing dataset shift issues artificially. In this paper, we suggest the use of a novel methodology called "Distribution optimally balanced SCV" (DOB–SCV) [31] when dealing with imbalanced datasets. This method attempts to minimize covariate shift by keeping data distribution as similar as possible between training and test folds by maximizing diversity on each fold and trying to keep all folds as resembling as possible to each other. The mechanism of this approach consists in selecting the k closest neighbours for a given instance and place them in different folds (with k being the number of total partitions), so that the data distribution between the training and test partitions remains as close as possible.

We must point out that neither SCV nor DOB–SCV can undoubtedly estimate the true classification error of a given model. In particular, there are several factors which may affect the output for unseen samples, and make some problems more difficult than others. Among others, we may stress uneven class distribution (as studied in this paper), the dimensionality of the problem and its relationship with the overlapping between the classes, and the presence of noise and/or outliers. However, we suggest that, by making the training and test partitions more similar between them, the use of DOB–SCV can guarantee a better average validation of the results. As pointed out previously, in this way we may avoid those classification errors which are due to dataset shift, especially those regarded to the minority class instances.

In order to evaluate the goodness and validity of the use of this new partitioning mechanism for imbalanced datasets, we develop a thorough empirical study by setting up an experimental framework which includes a set of sixty-six real-world problems from the KEEL dataset repository [3,4] (http://www.keel.es/dataset.php). We measure the performance of the classifiers based on its Area Under the Curve (AUC) metric [23] as suggested in imbalanced domains. Additionally, we study the significance of the results by the proper statistical tests as suggested in the literature [17,20]. Finally, we check the robustness of the DOB–SCV strategy using several well-known classifiers from different Machine Learning paradigms: decision trees [34], fuzzy rule based classification systems (FRBCS) [24], instance-based learning [1], and Support Vector Machines (SVMs) [12,15].

This study provides three significant contributions to the research community on classification with imbalanced data, namely:

1. We establish the motivation for the use of a new validation technique for avoiding dataset shift, which highly affects the performance in this scenario.
2. The goodness of this novel methodology is confirmed by means of a thorough experimental analysis. In this study, several algorithms from different paradigms were selected, showing better average performance estimates when using DOB–SCV.
3. Finally, we have concluded that the optimistic/pessimistic estimation of the performance also depends on the problem to be classified. In this way, the intrinsic data characteristics may have some degree of influence on the final results obtained by the classifier.

In order to carry out the study, this manuscript is organized as follows. First, Section 2 introduces the problem of imbalanced data. Next, Section 3 contains the main concepts that are developed in this work, i.e. the basis on validation techniques and the problem of covariate/dataset shift. Then, the experimental framework is presented in Section 4, whereas all the analysis of the results is shown along Section 5. Finally, Section 6 summarises and concludes the work.

2. Imbalanced datasets in classification

In this section, we will first introduce the problem of imbalanced datasets, describing its features and why is so difficult to learn in this classification scenario. Then, we will present how to address this problem, enumerating diverse approaches that can be applied to ease the discrimination of the minority (positive) and majority (negative) classes. Finally, we will discuss how to evaluate the performance of the results in this situation.

2.1. The problem of imbalanced datasets

The main property of this type of classification problem (in a binary context) is that the examples of one class outnumber the examples of the other one [11,38]. The minority classes are usually the most important concepts to be learnt, since they might be associated with exceptional and significant cases [42] or because the data acquisition of these examples is costly

[44]. Since most of the standard learning algorithms consider a balanced training set, this situation may cause the obtention of suboptimal classification models, i.e. a good coverage of the majority examples whereas the minority ones are misclassified more frequently [21,38].

Traditionally, the Imbalance Ratio (IR), i.e. the ratio between the majority and minority class examples [32], is the main hint to identify a set of problems which need to be addressed in a special way. Additionally, other data intrinsic characteristics that are related to this concept may include the overlapping between classes [26], lack of representative data [41], small disjuncts [33,43], dataset shift [29] and other issues which have interdependent effects with data distribution (imbalance).

The hitch here is that most learning algorithms aim to obtain a model with a high prediction accuracy and a good generalization capability. However, this inductive bias towards such a model poses a serious challenge to the classification of imbalanced data [38]. First, if the search process is guided by the standard accuracy rate, it benefits the covering of the majority examples; second, classification rules that predict the positive class are often highly specialized and thus their coverage is very low, hence they are discarded in favour of more general rules, i.e. those that predict the negative class. Furthermore, it is not easy to distinguish between noisy examples and positive class examples and they can be completely ignored by the classifier.

2.2. Addressing the imbalanced problem: preprocessing and cost-sensitive learning

A large number of approaches have been proposed to deal with the class imbalance problem [28], which can be categorized in three groups:

1. Data level solutions: the objective consists in rebalancing the class distribution by sampling the data space to diminish the effect caused by class imbalance, acting as an external approach [6,10,39].
2. Algorithmic level solutions: these solutions try to adapt several classification algorithms to reinforce the learning towards the positive class. Therefore, they can be defined as internal approaches that create new algorithms or modify existing ones to take the class imbalance problem into consideration [5,49].
3. Cost-sensitive solutions: this type of solutions incorporate approaches at the data level, at the algorithmic level, or at both levels jointly, considering higher costs for the misclassification of examples of the positive class with respect to the negative class, and therefore, trying to minimize higher cost errors [18,40,50].

The advantage of the data level solutions is that they are more versatile, since their use is independent of the classifier selected. Furthermore, we may preprocess all datasets before-hand in order to use them to train different classifiers. In this manner, we only need to prepare the data once. Furthermore, previous analysis on preprocessing methods with several classifiers have shown the goodness of the oversampling techniques [6].

The simplest approach, random oversampling, makes exact copies of existing instances, and therefore several authors agree that this method can increase the likelihood of occurring overfitting [6]. According to the previous fact, more sophisticated methods have been proposed based on the generation of synthetic samples. Among them, the "Synthetic Minority Over-sampling TEchnique" (SMOTE) [10] algorithm, whose main idea is to form new positive class examples by interpolating between several positive class examples that lie together, has become one of the most significant approaches in this area.

The positive class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbours. Depending upon the amount of over-sampling required, neighbours from the k nearest neighbours are randomly chosen. This process is illustrated in Fig. 1, where x_i is the selected point, x_{i1} to x_{i4} are some selected nearest neighbours and r_1 to r_4 the synthetic data points created by the randomised interpolation.

Synthetic samples are generated in the following way: take the difference between the feature vector (sample) under consideration and its nearest neighbour. Multiply this difference by a random number between 0 and 1, and add it to the feature vector under consideration. This causes the selection of a random point along the line segment between two specific features. This approach effectively forces the decision region of the positive class to become more general.

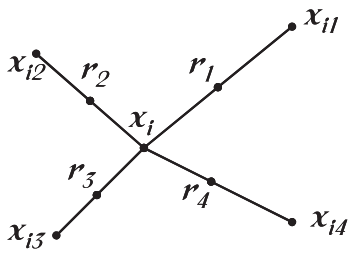


Fig. 1. An illustration of how to create the synthetic data points in the SMOTE algorithm.

Table 1
Confusion matrix for a two-class problem.

	Positive prediction	Negative prediction
Positive class	True Positive (TP)	False Negative (FN)
Negative class	False Positive (FP)	True Negative (TN)

2.3. Evaluation in imbalanced domains

The evaluation criteria is a key factor in both assessing the classification performance and guiding the classifier modelling. In a two-class problem, the confusion matrix (shown in Table 1) records the results of correctly and incorrectly recognized examples of each class.

Traditionally, accuracy rate (Eq. (1)) has been the most commonly used empirical measure. However, in the framework of imbalanced datasets, accuracy is no longer a proper measure, since it does not distinguish between the number of correctly classified examples of different classes. Hence, it may lead to erroneous conclusions, i.e., a classifier achieving an accuracy of 90% in a dataset with an IR value of 9, is not accurate if it classifies all examples as negatives.

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \tag{1}$$

In imbalanced domains, the evaluation of the classifiers' performance must be carried out using specific metrics to take into account the class distribution. Specifically, a well-known approach to produce an evaluation criteria in an imbalanced scenario is to use the Receiver Operating Characteristic (ROC) graphic [7]. This graphic allows to visualize the trade-off between the benefits (TP_{rate}) and costs (FP_{rate}), thus it evidences that any classifier cannot increase the number of true positives without also increasing the false positives. The Area Under the ROC Curve (*AUC*) [22] corresponds to the probability of correctly identifying which one of the two stimuli is noise and which one is signal plus noise. *AUC* provides a single measure of a classifier's performance for evaluating which model is better on average. Fig. 2 shows how to build the ROC space plotting on a two-dimensional chart the TP_{rate} (Y-axis) against the FP_{rate} (X-axis). Points in $(0,0)$ and $(1,1)$ are trivial classifiers where the predicted class is always the negative and positive respectively. On the contrary, $(0,1)$ point represents the perfect classification. The *AUC* measure is computed just by obtaining the area of the graphic:

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \tag{2}$$

3. Classifier evaluation techniques and the issue of dataset shift

As stated in the introduction of this work, the estimation of the performance of a classifier, via partitioning in training and test folds, is a necessary procedure in order to validate the results for a given experiment. However, the way this task is developed has a direct influence in the analysis of the obtained models. Specifically, the issue of dataset shift can occur when the distribution of the samples in training and test is quite different between them, leading to "overfitting".

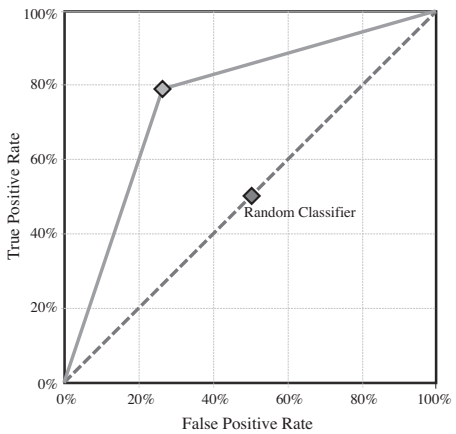


Fig. 2. Example of an ROC plot. Two classifiers' curves are depicted: the dashed line represents a random classifier, whereas the solid line is a classifier which is better than the random classifier.

In this section, we describe dataset shift in order to understand the nature of the problem we are dealing with. Next, we recall the standard and well-known SCV technique, and we identify its handicap for classification with imbalanced data. Finally, we present a recent methodology to alleviate this situation by a better organization of the instances among the different folds.

3.1. Dataset shift

The problem of dataset shift [2,8,36] is defined as the case where training and test data follow different distributions. This is a common problem that can affect all kind of classification problems, and it often appears due to sample selection bias issues. A mild degree of dataset shift is present in most real-world problems, but general classifiers are often capable of handling it without a severe performance loss.

There are three potential types of dataset shift:

1. *Prior Probability Shift:* It happens when the class distribution is different between the training and test sets [37]. In the most extreme example, the training set would not have a single example of a class, leading to a degenerate classifier. The problems caused by this kind of shift have already been studied, and they are commonly prevented by applying a SCV scheme [46].
2. *Covariate Shift:* In this case, it is the input attribute values that have different distributions between the training and test sets [36]. We focus on the impact of this type of shift for classification problems with imbalanced data.
3. *Concept Shift:* We refer to this problem when the relationship between the input and class variables changes [2,47], which presents the hardest challenge among the different types of dataset shift. In the specialized literature it is usually referred to as "Concept Drift" [27,45].

The dataset shift issue is specially relevant when dealing with imbalanced classification, because in highly imbalanced domains, the positive class is particularly sensitive to singular classification errors, due to the typically low number of examples it presents [29]. In the most extreme cases, a single misclassified example of the positive class can create a significant drop in performance.

For clarity, Figs. 3 and 4 present two examples of the influence of dataset shift in imbalanced classification. In the first case (Fig. 3), it is easy to see a separation between classes in the training set that carries over perfectly to the test set. However, in the second case (Fig. 4) it must be noted how some positive class examples in test are at the bottom and rightmost areas where there were not represented in the training set, leading to a gap between the training and test performance. These problems are represented in a two-dimensional space by means of a linear transformation of the inputs variables following the technique given by [29].

3.2. Cross-validation for classifier evaluation: distribution optimally balanced SCV

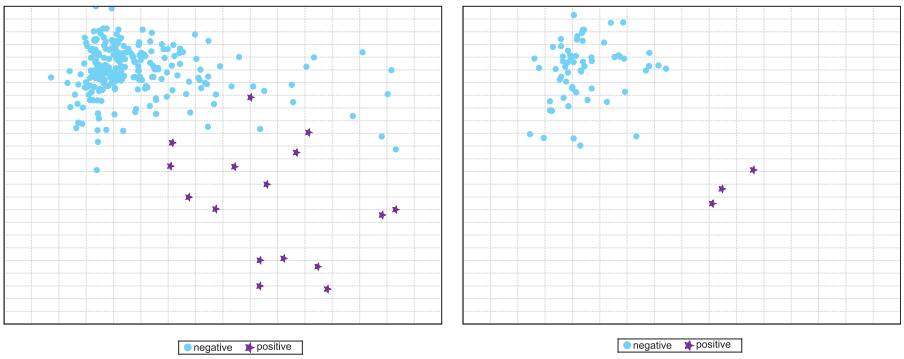
Cross-validation is a technique used for assessing how a classifier will perform when classifying new instances of the task at hand. One iteration of cross-validation involves partitioning a sample of data into two complementary subsets: training the classifier on one subset (called the training set) and testing its performance on the other subset (test set).

In k-fold cross-validation, the original sample is randomly partitioned into k subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the classifier, and the remaining $k-1$ subsamples are used as training data. The cross-validation process is then repeated k times, with each of the k subsamples used exactly once as the test data. The k results from the folds are then averaged to produce a single performance estimation.

The way the subsamples are assigned to each fold determines the impact of the final performance estimation in the validation stage. The most straightforward procedure is known as SCV, which works as follows: it counts how many samples of each class are there in the dataset, and distributes them evenly on the folds, so that each fold contains the same number of examples of each class. This avoids prior probability shift, because with an equal distribution class-wise on each fold, training and test set will have the same class distribution. However, this method does not take into account the covariates of the samples, so it can potentially generate covariate shift.

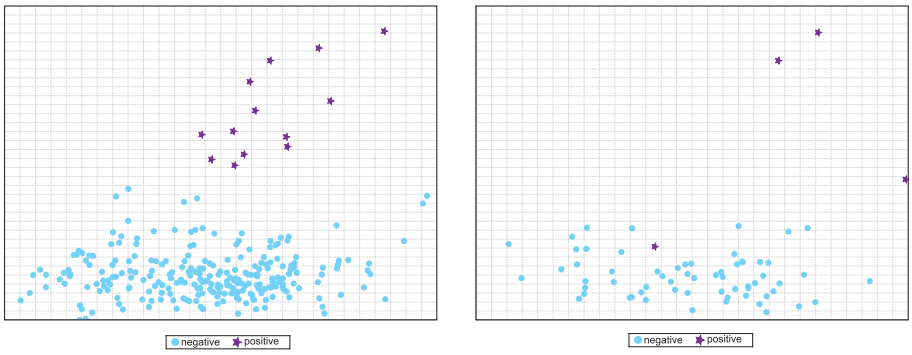
According to this fact, we consider a more sophisticated technique, known as DOB–SCV [31], which adds an extra consideration to the partitioning strategy as an attempt to alleviate the problem of covariate shift on top of preventing prior probability shift. The idea is that by assigning close-by examples to different folds, each fold will end up with enough representatives of every region, thus avoiding covariate shift.

This method is based on the Distribution-balanced SCV [52] and its pseudo-code is depicted in Algorithm 1. It picks a random unassigned example, and then finds its $k-1$ nearest unassigned neighbours of the same class. Once it has found them, it assigns each of those examples to a different fold. The process is repeated until there are no more examples of that class (when it gets to the last fold, it cycles and continues with the first one again). The whole process is repeated for each class.



(a) Training data. AUC = .9043 (b) Test data. AUC = 1.000

Fig. 3. Example of good behaviour (no dataset shift) in imbalanced domains: ecoli4 dataset, 5th partition.



(a) Training data. AUC = 1.000 (b) Test data. AUC = .8750

Fig. 4. Example of bad behaviour caused by dataset shift in imbalanced domains: ecoli4 dataset, 1st partition.

Algorithm 1. DOB–SCV partitioning method

for each class $c_j \in C$ **do**
  while count $(c_j) > 0$ **do**
    $e_0 \leftarrow$ randomly select an example of class c_j from D
    $e_i \leftarrow i$th closest example to e_0 of class c_j from D $(i = 1, \ldots, k-1)$
    $F_i \leftarrow F_i \bigcup e_i (i = 0, \ldots, k-1)$
    $D \leftarrow D \backslash e_i (i = 0, \ldots, k-1)$
  end while
end for

4. Experimental framework

In this section we first provide details of the real-world binary-class imbalanced problems chosen for the experiments (Section 4.1). Then, we will describe the learning algorithms selected for this study and their configuration parameters (Section 4.2). Finally, we present the statistical tests applied to compare the results obtained with the different classifiers (Section 4.3).

4.1. Benchmark data

There is no consensus in the research community on what threshold must be set up for a given dataset to suffer from the imbalance problem. In this paper, we consider a dataset to be imbalanced when the positive class has a distribution of examples below 40% of the number of instances that belong to the majority class, that is, if the ratio between the examples of the

Table 2
Summary of imbalanced datasets used.

Name	#Ex.	#Atts.	IR	Name	#Ex.	#Atts.	IR
Glass1	214	9	1.82	Glass04vs5	92	9	9.22
Ecoli0vs1	220	7	1.86	Ecoli0346vs5	205	7	9.25
Wisconsin	683	9	1.86	Ecoli0347vs56	257	7	9.28
Pima	768	8	1.90	Yeast05679vs4	528	8	9.35
Iris0	150	4	2.00	Ecoli067vs5	220	6	10.00
Glass0	214	9	2.06	Vowel0	988	13	10.10
Yeast1	1484	8	2.46	Glass016vs2	192	9	10.29
Vehicle1	846	18	2.52	Glass2	214	9	10.39
Vehicle2	846	18	2.52	Ecoli0147vs2356	336	7	10.59
Vehicle3	846	18	2.52	Led7digit02456789vs1	443	7	10.97
Haberman	306	3	2.68	Glass06vs5	108	9	11.00
Glass0123vs456	214	9	3.19	Ecoli01vs5	240	6	11.00
Vehicle0	846	18	3.23	Glass0146vs2	205	9	11.06
Ecoli1	336	7	3.36	Ecoli0147vs56	332	6	12.28
New-thyroid2	215	5	4.92	Cleveland0vs4	177	13	12.62
New-thyroid1	215	5	5.14	Ecoli0146vs5	280	6	13.00
Ecoli2	336	7	5.46	Ecoli4	336	7	13.84
Segment0	2308	19	6.01	Yeast1vs7	459	8	13.87
Glass6	214	9	6.38	Shuttle0vs4	1829	9	13.87
Yeast3	1484	8	8.11	Glass4	214	9	15.47
Ecoli3	336	7	8.19	Page-blocks13vs2	472	10	15.85
Page-blocks0	5472	10	8.77	Abalone9vs18	731	8	16.68
Ecoli034vs5	200	7	9.00	Glass016vs5	184	9	19.44
Yeast2vs4	514	8	9.08	Shuttle2vs4	129	9	20.50
Ecoli067vs35	222	7	9.09	Yeast1458vs7	693	8	22.10
Ecoli0234vs5	202	7	9.10	Glass5	214	9	22.81
Glass015vs2	172	9	9.12	Yeast2vs8	482	8	23.10
Yeast0359vs78	506	8	9.12	Yeast4	1484	8	28.41
Yeast02579vs368	1004	8	9.14	Yeast1289vs7	947	8	30.56
Yeast0256vs3789	1004	8	9.14	Yeast5	1484	8	32.78
Ecoli046vs5	203	6	9.15	Ecoli0137vs26	281	7	39.15
Ecoli01vs235	244	7	9.17	Yeast6	1484	8	39.15
Ecoli0267vs35	224	7	9.18	Abalone19	4174	8	128.87

majority and minority class is higher than 1.5. The data used in the study are summarized in Table 2, where we denote the number of examples (#Ex.), number of attributes (#Atts.) and IR. This table is in ascending order according to the IR.

As pointed out along this paper, the estimates of the AUC measure are obtained by means of a standard SCV and the DOB–SCV. The number of folds selected in both cases is 5. This value is set up with the aim of having enough positive class instances in the different folds, hence avoiding additional problems in the data distribution, especially for highly imbalanced datasets. Furthermore, we must point out that the original dataset partitions with 5-fold-cross-validation employed in this paper are available for download at the KEEL dataset repository [3] so that any interested researcher can use the same data for comparison.

4.2. Algorithms and parameters

In order to check the robustness of the DOB–SCV strategy, we have make use of several well-known classifiers from different Machine Learning paradigms: the C4.5 Decision Tree [34], the Chi et al. algorithm [13] as FRBCS [24], the well known *k*-NN algorithm [16] as instance-based learning method [1], and SVMs with both the Support Vector Machines with SMO optimization [15] and the Positive Definite Fuzzy Classifier (PDFC) [12]. Specifically, we have selected the following approaches as they are considered to be baseline algorithms in the field of Data Mining and they cover the widest used paradigms in classification. In this way, we can study the validity of our proposal within different types of classifiers, thus being able to generalize our extracted conclusions.

Next, we detail the parameter values for the different learning algorithms selected in this study, which have been set considering the recommendation of the corresponding authors:

1. **C4.5**
   For C4.5 we have set a confidence level of *0.25*, the minimum number of item-sets per leaf was set to 2 and the application of pruning was used to obtain the final tree.
2. **Chi et al.**
   We will apply a configuration consisting in product T-norm as conjunction operator, together with the Penalized Certainty Factor approach [25] for the rule weight, and winning rule as Fuzzy Reasoning Method [14]. Furthermore, we have selected the use of 5 labels per variable.

3. ***k*-NN**
   In this case we have selected 1 neighbour for determining the output class, applying the euclidean distance metric.
4. **SMO**
   The SMO algorithm was run using polynomial reference functions, with a value of 1 in the exponent of each kernel function and a penalty parameter of the error term of *1.0*.
5. **PDFC**
   The FRBCS part of this method applies a product T-norm as the fuzzy conjunction operator, addition for fuzzy rule aggregation, and centre of area defuzzification. For the SVM part we have chosen Gaussian functions for the kernels, with an internal parameter of *0.25* and the weight of the classification error set to *100.0*.

Regarding the SMOTE preprocessing technique, we will consider the *5-nearest neighbours of the positive class* to generate the synthetic samples, and *balancing both classes to the 50% distribution*.

We must also point out that all these algorithms are available within the KEEL software tool [4].

4.3. Statistical tests for performance comparison

The goodness of a given approach cannot be only measured in terms of the improvement for the mean performance. Significant differences must be found among the different algorithms for concluding the superior behaviour of the one that achieves the highest average result.

For this reason, in this paper we use the hypothesis testing techniques to provide statistical support for the analysis of the results [19,35]. Specifically, we will use non-parametric tests, due to the fact that the initial conditions that guarantee the reliability of the parametric tests may not be satisfied, causing the statistical analysis to lose credibility with these type of tests [17].

We apply the Wilcoxon signed-rank test [35] as a non-parametric statistical procedure for performing pairwise comparisons between two algorithms, as the analogous of the paired *t*-test. This procedure computes the differences between the performance scores of the two classifiers on i^{th} out of N_{ds} datasets. The differences are ranked according to their absolute values, from smallest to largest, and average ranks are assigned in case of ties. We call R^+ the sum of ranks for the datasets on which the second algorithm outperformed the first, and R^- the sum of ranks for the opposite. Let T be the smallest of the sums, $T = min(R^+, R^-)$. If T is less than or equal to the value of the distribution of Wilcoxon for N_{ds} degrees of freedom (Table B.12 in [51]), the null hypothesis of equality of means is rejected.

This statistical test allows us to know whether a hypothesis of comparison of means could be rejected at a specified level of significance α. It is also very interesting to compute the *p*-value associated to each comparison, which represents the lowest level of significance of a hypothesis that results in a rejection. In this manner, we can know whether two algorithms are significantly different and how different they are.

Non-parametrical tests are suggested in the studies presented in [17,19,20], where its use in the field of machine learning is highly recommended. Any interested reader can find additional information on the Website http://sci2s.ugr.es/sicidm/.

5. Experimental study

This section is devoted to identify the possible differences regarding the estimation of the performance with the standard SCV and the suggested DOB–SCV for imbalanced datasets.

Table 3 shows the average results for the five algorithms selected for our study, namely C4.5, FRBCS (Chi et al.), 1-NN, SMO and PDFC, grouped with respect to the IR. We must recall that, in order to address imbalance, these results are computed using SMOTE as preprocessing technique.

For each classification method, three values are given: first the average AUC performance together with its standard variation obtained in the test partitions for the SCV technique, then the average performance for DOB–SCV, and finally the relative difference between both values, i.e. $\frac{AUC_{DOB-SCV} - AUC_{SCV}}{AUC_{SCV}}$. In this manner, if the value is positive it means that the estimation of the performance for DOB–SCV is more optimistic than SCV; if the value is negative it refers to the contrary case; and the

Table 3
Average test results with AUC metric and percentage differences for the SCV and DOB–SCV techniques.

Algorithm	IR < 9			IR > 9			All		
	SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.
C4.5	.8597 ± .0357	.8698 ± .0393	1.28	.8133 ± .0844	.8309 ± .0751	2.83	.8288 ± .0681	.8439 ± .0632	2.32
Chi	.8151 ± .0352	.8187 ± .0380	0.51	.7698 ± .1041	.7781 ± .0909	1.24	.7849 ± .0811	.7916 ± .0733	1.00
k-NN	.8478 ± .0342	.8616 ± .0340	1.96	.8272 ± .0937	.8395 ± .0855	1.74	.8341 ± .0739	.8468 ± .0683	1.81
SMO	.8573 ± .0317	.8644 ± .0253	0.96	.8425 ± .0695	.8427 ± .0606	0.23	.8474 ± .0569	.8500 ± .0488	0.47
PDFC	.8877 ± .0293	.8901 ± .0263	0.34	.8608 ± .0819	.8672 ± .0708	0.86	.8698 ± .0644	.8749 ± .0560	0.69

Table 4
Detailed test results with AUC metric and percentage differences for the SCV and DOB–SCV techniques. Values are grouped by classification algorithm.

Dataset	IR	C4.5			Chi			*k*-NN			SMO			PDFC		
		SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.
Glass1	1.82	.7577 ± .0379	.7416 ± .0413	−2.12	.6788 ± .0663	.6567 ± .0670	−3.26	.7738 ± .0561	.8000 ± .0454	3.38	.5692 ± .0676	.6091 ± .0133	7.01	.7072 ± .0259	.7303 ± .0215	3.26
Ecoli0vs1	1.86	.9761 ± .0190	.9806 ± .0178	0.46	.9570 ± .0498	.9516 ± .0320	−0.57	.9626 ± .0302	.9704 ± .0129	0.80	.9796 ± .0219	.9808 ± .0175	0.12	.9831 ± .0167	.9841 ± .0156	0.10
Wisconsin	1.86	.9545 ± .0199	.9585 ± .0116	0.42	.5734 ± .0215	.5699 ± .0284	−0.62	.9624 ± .0177	.9655 ± .0118	0.33	.9706 ± .0103	.9728 ± .0061	0.23	.9568 ± .0051	.9566 ± .0146	−0.01
Pima	1.90	.7145 ± .0388	.7451 ± .0366	4.27	.6714 ± .0251	.7010 ± .0545	4.41	.6808 ± .0505	.6940 ± .0432	1.93	.7412 ± .0397	.7424 ± .0240	0.16	.7508 ± .0351	.7482 ± .0515	−0.35
Iris0	2.00	.9900 ± .0224	.9900 ± .0224	0.00	1.0000 ± .0000	1.0000 ± .0000	0.00	1.0000 ± .0000	1.0000 ± .0000	0.00	1.0000 ± .0000	1.0000 ± .0000	0.00	1.0000 ± .0000	1.0000 ± .0000	0.00
Glass0	2.06	.7856 ± .0234	.7709 ± .1189	−1.87	.6826 ± .0223	.6865 ± .0333	0.56	.8595 ± .0598	.8171 ± .0768	−4.93	.7117 ± .0298	.7183 ± .0123	0.93	.7576 ± .0821	.7722 ± .0650	1.92
Yeast1	2.46	.7113 ± .0295	.7117 ± .0424	0.05	.6994 ± .0142	.6974 ± .0374	−0.28	.6533 ± .0318	.6596 ± .0121	0.97	.7038 ± .0273	.7146 ± .0372	1.54	.7152 ± .0347	.7176 ± .0179	0.34
Vehicle1	2.52	.7468 ± .0125	.7222 ± .0451	−3.30	.6348 ± .0185	.6858 ± .0324	8.04	.6323 ± .0204	.6862 ± .0395	8.52	.7470 ± .0254	.7588 ± .0245	1.59	.8732 ± .0193	.8598 ± .0180	−1.53
Vehicle2	2.52	.9476 ± .0160	.9547 ± .0174	0.76	.8735 ± .0312	.8648 ± .0166	−1.00	.9539 ± .0202	.9299 ± .0149	−2.52	.9287 ± .0103	.9247 ± .0202	−0.42	.9811 ± .0074	.9806 ± .0097	−0.06
Vehicle3	2.52	.7015 ± .0281	.7290 ± .0460	3.92	.7212 ± .0123	.6946 ± .0225	−3.69	.6835 ± .0235	.6562 ± .0642	−3.99	.7282 ± .0376	.7376 ± .0370	1.30	.8401 ± .0152	.8329 ± .0237	−0.85
Haberman	2.68	.6309 ± .0407	.6521 ± .0227	3.37	.6185 ± .0266	.6123 ± .0935	−0.99	.5394 ± .0525	.6096 ± .0869	13.01	.6161 ± .0612	.6199 ± .0460	0.61	.6120 ± .0587	.6256 ± .0588	2.23
Glass0123vs456	3.19	.8832 ± .0605	.9256 ± .0363	4.80	.8640 ± .0140	.8662 ± .0655	0.25	.9224 ± .0154	.9395 ± .0393	1.85	.8819 ± .0714	.9173 ± .0331	4.02	.9292 ± .0512	.9374 ± .0386	0.88
Vehicle0	3.23	.9143 ± .0237	.9465 ± .0073	3.52	.8495 ± .0160	.8697 ± .0209	2.38	.9106 ± .0191	.9301 ± .0245	2.14	.9562 ± .0159	.9587 ± .0128	0.26	.9764 ± .0095	.9813 ± .0065	0.50
Ecoli1	3.36	.9162 ± .0485	.8661 ± .0358	−5.47	.8791 ± .0487	.8793 ± .0190	0.02	.8298 ± .0783	.8934 ± .0285	7.66	.8933 ± .0452	.8931 ± .0133	−0.03	.8967 ± .0546	.8854 ± .0153	−1.26
New-thyroid2	4.92	.9631 ± .0456	.9833 ± .0181	2.10	.9659 ± .0612	.9746 ± .0275	0.90	.9774 ± .0279	.9690 ± .0325	−0.85	.9774 ± .0296	.9944 ± .0076	1.75	.9917 ± .0076	.9917 ± .0124	0.00
New-thyroid1	5.14	.9802 ± .0371	.9690 ± .0473	−1.13	.9548 ± .0859	.9603 ± .0664	0.58	.9774 ± .0279	.9806 ± .0124	0.32	.9861 ± .0170	.9889 ± .0152	0.28	.9944 ± .0076	.9917 ± .0124	−0.28
Ecoli2	5.46	.8921 ± .0715	.8834 ± .0500	−0.97	.9170 ± .0490	.9061 ± .0400	−1.19	.9343 ± .0505	.9272 ± .0414	−0.77	.9085 ± .0469	.9046 ± .0427	−0.43	.9381 ± .0419	.9311 ± .0405	−0.75
Segment0	6.01	.9927 ± .0060	.9912 ± .0076	−0.15	.9590 ± .0121	.9649 ± .0066	0.61	.9949 ± .0066	.9934 ± .0038	−0.15	.9917 ± .0090	.9917 ± .0054	0.00	.9960 ± .0033	.9990 ± .0017	0.30
Glass6	6.38	.8450 ± .0750	.8896 ± .0839	5.28	.7969 ± .0679	.8396 ± .0834	5.36	.8686 ± .0867	.9365 ± .0664	7.82	.9057 ± .0552	.9365 ± .0719	3.40	.8938 ± .0813	.9176 ± .0714	2.66
Yeast3	8.11	.8869 ± .0344	.9086 ± .0363	2.45	.8942 ± .0337	.8881 ± .0281	−0.68	.8607 ± .0134	.8693 ± .0297	1.00	.9040 ± .0128	.9003 ± .0314	−0.41	.9224 ± .0213	.9301 ± .0246	0.83
Ecoli3	8.19	.7755 ± .0787	.8677 ± .1067	11.90	.8665 ± .0801	.8681 ± .0423	0.19	.7777 ± .0482	.8139 ± .0469	4.65	.8874 ± .0418	.8758 ± .0678	−1.31	.8798 ± .0554	.8797 ± .0462	−0.01
Page-blocks0	8.77	.9484 ± .0153	.9472 ± .0140	−0.12	.8744 ± .0185	.8752 ± .0189	0.09	.8953 ± .0155	.9135 ± .0147	2.03	.8729 ± .0215	.8774 ± .0169	0.52	.9335 ± .0101	.9300 ± .0122	−0.37
Ecoli034vs5	9.00	.8583 ± .0806	.8694 ± .1118	1.29	.8194 ± .1343	.8222 ± .1055	0.34	.8472 ± .1361	.8639 ± .1323	1.97	.8944 ± .1037	.9000 ± .0669	0.62	.8833 ± .1139	.8889 ± .1080	0.63
Yeast2vs4	9.08	.8620 ± .0589	.8716 ± .0358	1.11	.8607 ± .0492	.8727 ± .0282	1.39	.8807 ± .0655	.8905 ± .0504	1.11	.8863 ± .0287	.8963 ± .0244	1.13	.9154 ± .0608	.9201 ± .0344	0.51
Ecoli067vs35	9.09	.8125 ± .2097	.8225 ± .0945	1.23	.7925 ± .1660	.7850 ± .1084	−0.95	.8625 ± .1495	.8675 ± .1037	0.58	.8550 ± .1509	.8500 ± .0824	−0.58	.8650 ± .1687	.8800 ± .0873	1.73
Ecoli0234vs5	9.10	.8974 ± .1051	.8528 ± .0871	−4.97	.8114 ± .1577	.8725 ± .1043	7.53	.8530 ± .1261	.8808 ± .1102	3.26	.8946 ± .1109	.9029 ± .0972	0.92	.9056 ± .1135	.8862 ± .0964	−2.15
Glass015vs2	9.12	.7444 ± .1152	.6411 ± .0694	−13.87	.5583 ± .0848	.5126 ± .1752	−8.18	.6573 ± .1287	.6290 ± .1018	−4.29	.5344 ± .0400	.5737 ± .0968	7.34	.8043 ± .1182	.7793 ± .1021	−3.11
Yeast0359vs78	9.12	.7222 ± .0537	.7022 ± .0874	−2.77	.7040 ± .0631	.7063 ± .0527	0.32	.7543 ± .0384	.7188 ± .0856	−4.72	.7428 ± .0415	.7495 ± .0635	0.90	.7170 ± .0377	.7028 ± .0786	−1.99
Yeast02579vs368	9.14	.9171 ± .0164	.9044 ± .0325	−1.39	.8871 ± .0380	.8813 ± .0421	−0.65	.9044 ± .0282	.8927 ± .0493	−1.28	.9035 ± .0366	.9027 ± .0336	−0.09	.9021 ± .0319	.9037 ± .0395	0.19
Yeast0256vs3789	9.14	.7543 ± .0242	.7771 ± .0585	3.02	.7798 ± .0763	.7837 ± .0233	0.49	.7807 ± .0556	.8068 ± .0475	3.33	.7940 ± .0510	.8095 ± .0376	1.96	.8189 ± .0528	.8142 ± .0274	−0.58
Ecoli046vs5	9.15	.8729 ± .0993	.8342 ± .1094	−4.44	.8394 ± .1434	.8533 ± .1645	1.65	.8642 ± .1427	.8918 ± .1073	3.20	.8979 ± .1086	.8978 ± .1092	−0.01	.8507 ± .0926	.9086 ± .1061	6.81
Ecoli01vs235	9.17	.8041 ± .1660	.8377 ± .1359	4.18	.7441 ± .0805	.8209 ± .0691	1.32	.8286 ± .1507	.8850 ± .1069	6.80	.8577 ± .0923	.8764 ± .0838	2.17	.8868 ± .1512	.9214 ± .0720	3.90
Ecoli0267vs35	9.18	.7704 ± .1082	.8606 ± .0869	11.71	.7753 ± .0752	.7881 ± .1398	1.65	.8976 ± .0985	.8928 ± .0918	−0.53	.8731 ± .0776	.8730 ± .0840	−0.01	.8426 ± .1085	.8804 ± .0533	4.49
Glass04vs5	9.22	.9816 ± .0168	.9706 ± .0294	−1.12	.7210 ± .1989	.7224 ± .1422	0.20	.9691 ± .0383	.9206 ± .1156	−5.01	.9629 ± .0408	.9581 ± .0155	−0.50	.9636 ± .0254	.9706 ± .0416	0.72
Ecoli0346vs5	9.25	.8703 ± .0517	.8784 ± .1196	0.93	.8568 ± .0986	.8176 ± .1156	−4.57	.8838 ± .0986	.8561 ± .0933	−3.13	.8953 ± .0587	.8953 ± .0589	0.00	.9169 ± .0707	.9115 ± .0627	−0.59
Ecoli0347vs56	9.28	.8368 ± .1514	.8992 ± .0443	7.45	.8196 ± .1107	.8306 ± .1124	1.34	.8834 ± .1215	.8764 ± .0930	−0.79	.9191 ± .0888	.8905 ± .0813	−3.11	.9055 ± .0838	.9185 ± .0809	1.44
Yeast05679vs4	9.35	.7682 ± .1009	.7954 ± .0827	3.55	.7989 ± .0625	.8011 ± .0481	0.28	.7753 ± .0599	.8003 ± .0597	3.23	.7885 ± .0849	.8005 ± .0182	1.51	.7900 ± .0930	.8014 ± .0392	1.45
Ecoli067vs5	1.00	.8250 ± .0862	.8875 ± .0690	7.58	.8275 ± .0958	.7875 ± .0631	−4.83	.8675 ± .0577	.8800 ± .0677	1.44	.8675 ± .0855	.8450 ± .0841	−2.59	.8700 ± .0473	.8650 ± .1088	−0.57
Vowel0	1.10	.9433 ± .0483	.9750 ± .0155	3.36	.9789 ± .0183	.9933 ± .0015	1.47	1.0000 ± .0000	.9989 ± .0015	−0.11	.9566 ± .0117	.9599 ± .0092	0.35	.9989 ± .0015	.9994 ± .0012	0.06
Glass016vs2	1.29	.6367 ± .1255	.6752 ± .1478	6.06	.6002 ± .0841	.5140 ± .1171	−14.36	.6814 ± .1793	.6976 ± .1674	2.38	.5379 ± .1120	.5819 ± .0771	8.19	.7605 ± .1208	.7769 ± .1316	2.16
Glass2	1.39	.5424 ± .1401	.7498 ± .1155	38.25	.5206 ± .1120	.6241 ± .0982	19.89	.6447 ± .0987	.7331 ± .1316	13.72	.5985 ± .1570	.5989 ± .0841	0.07	.7688 ± .1486	.7789 ± .1081	1.31
Ecoli0147vs2356	1.59	.8461 ± .0453	.8426 ± .0669	−0.41	.7894 ± .0606	.8043 ± .0728	1.89	.8507 ± .0309	.8857 ± .1008	4.11	.8844 ± .0767	.8862 ± .0461	0.20	.9025 ± .0542	.8823 ± .0158	−2.23
Led7digit02456789vs1	1.97	.8832 ± .0962	.8207 ± .0995	−7.08	.8302 ± .0749	.7983 ± .0810	−3.84	.8108 ± .0333	.8652 ± .0432	6.70	.8875 ± .0531	.8248 ± .0549	−7.07	.8852 ± .0923	.8611 ± .0954	−2.73
Glass06vs5	11.00	.9147 ± .1186	.9600 ± .0285	4.95	.7500 ± .2215	.7850 ± .1876	4.67	.9400 ± .1207	.9200 ± .1242	−2.13	.9439 ± .0344	.9492 ± .0260	0.56	.9745 ± .0358	.9597 ± .0133	−1.51
Ecoli01vs5	11.00	.8227 ± .1074	.8523 ± .0114	3.59	.8386 ± .1447	.8500 ± .1445	1.36	.8545 ± .1525	.8909 ± .0973	4.26	.8932 ± .0756	.8977 ± .0663	0.51	.8795 ± .1018	.9114 ± .0660	3.62
Glass0146vs2	11.06	.7564 ± .1089	.7361 ± .1509	−2.68	.5146 ± .1054	.5527 ± .1197	7.39	.6453 ± .0884	.7445 ± .1301	15.37	.6157 ± .0732	.6185 ± .0496	0.45	.8029 ± .1359	.7838 ± .0650	−2.38
Ecoli0147vs56	12.28	.8641 ± .0565	.8474 ± .0425	−1.93	.8441 ± .1129	.8458 ± .0535	0.19	.8756 ± .0622	.8740 ± .0717	−0.19	.9093 ± .0353	.8928 ± .0760	−1.82	.8907 ± .0755	.9124 ± .0831	2.44

(*continued on next page*)

Table 4 (*continued*)

Dataset	IR	C4.5			Chi			*k*-NN			SMO			PDFC		
		SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.	SCV	DOB–SCV	% Diff.
Cleveland0vs4	12.62	.7210 ± .1259	.7719 ± .1180	7.05	.1188 ± .0538	.1188 ± .0580	−0.08	.8543 ± .1430	.7042 ± .0652	−17.57	.9076 ± .0619	.9010 ± .0932	−0.72	.8929 ± .0765	.8188 ± .1538	−8.30
Ecoli0146vs5	13.00	.8981 ± .0975	.8538 ± .0797	−4.93	.8481 ± .1215	.8712 ± .1330	2.72	.8481 ± .1171	.9154 ± .1120	7.94	.8846 ± .0947	.8962 ± .1101	1.30	.8750 ± .1088	.9096 ± .1127	3.96
Ecoli4	13.84	.8044 ± .1388	.8980 ± .0732	11.64	.9230 ± .0813	.9152 ± .0771	−0.85	.9171 ± .0689	.9608 ± .0527	4.77	.9481 ± .0590	.8997 ± .0632	−5.10	.9060 ± .0724	.9012 ± .0695	−0.53
Yeast1vs7	13.87	.7064 ± .0671	.6711 ± .1027	−5.00	.6524 ± .1047	.6671 ± .0913	2.26	.7479 ± .1279	.6610 ± .0746	−11.62	.7691 ± .0642	.7477 ± .0431	−2.78	.6881 ± .0521	.7071 ± .0832	2.76
Shuttle0vs4	13.87	.9997 ± .0007	.9991 ± .0008	−0.06	.9872 ± .0117	.9874 ± .0281	0.02	.9960 ± .0089	.9957 ± .0088	−0.03	.9960 ± .0089	.9960 ± .0089	0.00	.9960 ± .0089	.9960 ± .0089	0.00
Glass4	15.47	.8508 ± .0935	.8986 ± .1376	5.61	.8618 ± .1105	.8762 ± .1459	1.67	.8917 ± .1162	.9085 ± .1491	1.88	.8928 ± .1161	.8713 ± .1429	−2.41	.9251 ± .1052	.9344 ± .0786	1.01
Page-blocks13vs4	15.85	.9955 ± .0047	.9565 ± .0752	−3.91	.8928 ± .1067	.8684 ± .0810	−2.74	.9977 ± .0051	.9876 ± .0074	−1.01	.7223 ± .1226	.8096 ± .0648	12.09	.9752 ± .0124	.9741 ± .0129	−0.11
Abalone9vs18	16.68	.6201 ± .0514	.7854 ± .0794	26.66	.6744 ± .0988	.6937 ± .0938	2.86	.6820 ± .0814	.7457 ± .0669	9.34	.8458 ± .0564	.7977 ± .0524	−5.68	.8969 ± .0227	.8373 ± .0577	−6.65
Glass016vs5	19.44	.9714 ± .0143	.9686 ± .0120	−0.29	.8486 ± .2191	.8514 ± .1435	0.34	.8771 ± .2191	.9329 ± .1118	6.35	.9343 ± .0329	.9371 ± .0192	0.31	.8771 ± .2274	.9214 ± .1229	5.05
Shuttle2vs4	2.50	.9958 ± .0093	.9877 ± .0185	−0.82	.8838 ± .2160	.8840 ± .2161	0.02	1.0000 ± .0000	.9958 ± .0093	−0.42	.9960 ± .0089	.9960 ± .0089	0.00	.9960 ± .0089	.9960 ± .0089	0.00
Yeast1458vs7	22.10	.5230 ± .0170	.5889 ± .0623	12.59	.5713 ± .0830	.6061 ± .0390	6.10	.6390 ± .0778	.6290 ± .0625	−1.56	.6570 ± .0612	.6539 ± .0745	−0.46	.6569 ± .0439	.7024 ± .0548	6.92
Glass5	22.81	.8829 ± .1331	.9829 ± .0139	11.33	.7463 ± .2052	.8439 ± .1281	13.07	.8829 ± .2148	.9232 ± .1182	4.56	.9341 ± .0318	.9390 ± .0228	0.52	.8732 ± .1145	.9256 ± .0984	6.01
Yeast2vs8	23.10	.8066 ± .1122	.7490 ± .0980	−7.13	.8066 ± .0694	.7099 ± .0566	−12.00	.8055 ± .1425	.7501 ± .1096	−6.88	.7664 ± .0960	.7663 ± .0495	−0.01	.7924 ± .1055	.7892 ± .0713	−0.41
Yeast4	28.41	.7004 ± .0565	.7823 ± .0786	11.69	.8325 ± .0239	.8303 ± .0209	−0.27	.7242 ± .0593	.7668 ± .0899	5.88	.8217 ± .0430	.8352 ± .0629	1.64	.8090 ± .0774	.8155 ± .0819	0.80
Yeast1289vs7	3.56	.7051 ± .0697	.6037 ± .0724	−14.38	.6770 ± .0853	.7027 ± .0665	3.80	.6444 ± .0713	.6503 ± .0877	0.92	.7216 ± .0514	.7227 ± .0713	0.15	.6964 ± .0938	.7126 ± .0506	2.31
Yeast5	32.78	.9337 ± .0400	.9389 ± .0266	0.56	.9372 ± .0272	.9465 ± .0256	1.00	.9326 ± .0413	.9514 ± .0333	2.01	.9656 ± .0068	.9653 ± .0069	−0.04	.9611 ± .0290	.9396 ± .0302	−2.24
Ecoli0137vs26	39.15	.8136 ± .2171	.8780 ± .1215	7.92	.7917 ± .1981	.8598 ± .1340	8.60	.8281 ± .2087	.8836 ± .1263	6.69	.8490 ± .1969	.8489 ± .1209	−0.01	.8118 ± .1957	.8744 ± .1266	7.72
Yeast6	39.15	.8280 ± .1277	.7996 ± .1199	−3.44	.8820 ± .0855	.8796 ± .0488	−0.27	.7998 ± .1200	.8361 ± .1274	4.54	.8751 ± .0712	.8744 ± .0494	−0.08	.8684 ± .0610	.8562 ± .0730	−1.41
Abalone19	128.87	.5203 ± .0443	.5827 ± .0811	11.99	.6748 ± .1077	.6976 ± .0424	3.38	.5176 ± .0385	.5763 ± .0653	11.34	.7894 ± .0463	.7908 ± .0729	0.18	.6777 ± .0529	.7280 ± .1019	7.42
Average		.8288 ± .0681	.8439 ± .0632	2.32	.7849 ± .0811	.7916 ± .0733	1.00	.8341 ± .0739	.8468 ± .0683	1.81	.8474 ± .0569	.8500 ± .0488	0.47	.8698 ± .0644	.8749 ± .0560	0.69

Table 5
Wilcoxon's tests to compare the results with the DOB–SCV versus the standard SCV. R^+ corresponds to the sum of the ranks for the DOB–SCV partitioning approach and and R^- to the original SCV partitioning.

Comparison	R^+	R^-	p-value
C4.5[DOB–SCV] vs C4.5[SCV]	1391	754	0.0371
Chi[DOB–SCV] vs Chi[SCV]	1411	734	0.0267
k-NN[DOB–SCV] vs k-NN[SCV]	1536	609	0.0024
SMO[DOB–SCV] vs SMO[SCV]	1395	816	0.0639
PDFC[DOB–SCV] vs PDFC[SCV]	1366	845	0.0955

higher the obtained number, the most significant the selection of the validation approach is. Additionally, we show the detailed test results for all datasets in Table 4.

From these tables of results we may observe that for all five algorithms, the DOB–SCV validation technique achieves a higher estimation of the performance for most datasets, therefore being more robust for analyzing the quality of the models learned in imbalanced data.

Furthermore, we must point out that the degree of imbalance of the dataset has a direct impact on the diverse results over the different folds in the obtained results, i.e. the higher the IR is, the greater the differences between the standard SCV and the DOB–SCV are. In addition to the former, the standard deviation computation supports this perception: these values for both partitioning techniques are similar when the degree of imbalance is low; however, when the IR is higher we may observe that the standard deviation is much higher in contrast with low imbalanced datasets. Additionally, DOB–SCV has lower standard deviation values than SCV, therefore sustaining the reduction of the gap between training and test partitions.

This issue may arise due to the fact that, the lower the number of positive instances we have in a dataset with respect to the negative ones, the more significant is to maintain the data distribution to avoid the differences in performance between training and test.

The characteristics of specific datasets do not pose a source of knowledge when trying to observe if there is a group of them where DOB–SCV performs better than SCV. In general, DOB–SCV obtains a better performance for most of the algorithms for each dataset, however, only few of the datasets considered are able to provide a clear trend for all the algorithms: the cases where DOB–SCV obtains a better estimation than SCV (for instance, *Abalone*19 or *Glass*2) are more numerous than the contrary case (*Ecoli*2 or *Yeast*2*vs*8) and the improvement is much greater than the loss.

When trying to find a group of data with the highest differences between DOB–SCV and SCV, it is not possible to do so without also considering the algorithm underneath. For instance, if we try to observe where the greatest improvements or losses are obtained for each algorithm, we realize that the datasets obtained for one algorithm are completely different from the datasets obtained for the rest.

In order to give statistical support to the findings previously extracted, we will carry out a Wilcoxon test to compare both validation techniques with the five classification algorithms. This analysis is shown in Table 5 where the algorithms are compared by rows.

The conclusions from this test are clear, from which significant differences are found between DOB–SCV and SCV in all cases with a low p-value. Furthermore, the higher sums of the ranks for DOB–SCV tell us about the goodness of this approach.

To summarize, we must stress that DOB–SCV is a suitable methodology for contrasting the performance of the classification algorithms in imbalanced data. When the distribution of the classes is skewed, using standard estimation models may lead to misleading conclusions on the quality of the prediction. The proposed use of this model addresses the handicap of losing the generalization ability because of the way data is distributed among the different folds.

6. Concluding remarks

In this work we have proposed the use of a novel partition-based methodology, named as DOB–SCV, which aims at obtaining a better estimation of a classifier's performance by carrying out an heterogeneous organization of the instances of the classes among the different folds.

We have identified this validation technique as a very suitable procedure in the framework of imbalanced datasets. It is straightforward to realize that, in the case that one of the classes of the problem contains a fewer number of examples, and regarding to the evaluation metrics used in this scenario, introducing covariate shift between training and test will unequivocally lead to high differences in performance in the learning and validation stages.

The stable performance estimation of DOB–SCV has been contrasted versus the classical k-fold SCV, detecting significant differences between both techniques for several classifiers often used in imbalanced tasks such as C4.5, FRBCSs, k-NN and SVMs. We must highlight that avoiding different data distribution inside each fold will allow researchers on imbalanced data to concentrate their efforts on designing new learning models based only on the skewed data, rather than seeking for complex solutions when trying to overcome the gaps between training and test results. Nevertheless, neither SCV nor DOB–SCV can unequivocally guarantee to obtain the best estimate of the true error for a given problem. This can only be achieved by having infinite data or, at least, that the input data covers the whole problem space, which is not usually the case.

Acknowledgments

This work was partially supported by the Spanish Ministry of Science and Technology under Project TIN2011-28488 and the Andalusian Research Plans P11-TIC-7765 and P10-TIC-6858. V. López holds a FPU scholarship from Spanish Ministry of Education.

References

[1] D.W. Aha, D. Kibler, M.K. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.
[2] R. Alaiz-Rodríguez, N. Japkowicz, Assessing the impact of changing environments on classifier performance, in: Proceedings of the 21st Canadian Conference on Advances in Artificial Intelligence (CCAI'08), Springer-Verlag, Berlin, Heidelberg, 2008.
[3] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, Journal of Multi-Valued Logic and Soft Computing 17 (2–3) (2011) 255–287.
[4] J. Alcalá;-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Computing 13 (2009) 307–318.
[5] R. Barandela, J.S. Sánchez, V. García, E. Rangel, Strategies for learning in class imbalance problems, Pattern Recognition 36 (3) (2003) 849–851.
[6] G.E.A.P.A. Batista, R.C. Prati, M.C. Monard, A study of the behaviour of several methods for balancing machine learning training data, SIGKDD Explorations 6 (1) (2004) 20–29.
[7] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognition 30 (7) (1997) 1145–1159.
[8] J.Q. Candela, M. Sugiyama, A. Schwaighofer, N.D. Lawrence, Dataset Shift in Machine Learning, The MIT Press, 2009.
[9] J.R. Cano, F. Herrera, M. Lozano, Evolutionary stratified training set selection for extracting classification rules with trade off precision-interpretability, Data and Knowledge Engineering 60 (2007) 90–108.
[10] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, Journal of Artificial Intelligent Research 16 (2002) 321–357.
[11] N.V. Chawla, N. Japkowicz, A. Kotcz, Editorial: special issue on learning from imbalanced data sets, SIGKDD Explorations 6 (1) (2004) 1–6.
[12] Y. Chen, J. Wang, Support vector learning for fuzzy rule-based classification systems, IEEE Transactions on Fuzzy Systems 11 (6) (2003) 716–728.
[13] Z. Chi, H. Yan, T. Pham, Fuzzy Algorithms with Applications to Image Processing and Pattern Recognition, World Scientific, 1996.
[14] O. Cordón, M.J. del Jesus, F. Herrera, A proposal on reasoning methods in fuzzy rule-based classification systems, International Journal of Approximate Reasoning 20 (1) (1999) 21–45.
[15] C. Cortes, V. Vapnik, Support vector networks, Machine Learning 20 (1995) 273–297.
[16] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1967) 21–27.
[17] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.
[18] P. Domingos, Metacost: a general method for making classifiers cost-sensitive, in: Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD'99), 1999.
[19] S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Computing 13 (10) (2009) 959–977.
[20] S. García, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, Journal of Machine Learning Research 9 (2008) 2607–2624.
[21] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Transactions on Knowledge and Data Engineering 21 (9) (2009) 1263–1284.
[22] J. Huang, C.X. Ling, Using AUC and accuracy in evaluating learning algorithms, IEEE Transactions on Knowledge and Data Engineering 17 (3) (2005) 299–310.
[23] Y.-M. Huang, C.-M. Hung, H.C. Jiau, Evaluation of neural networks and data mining methods on a credit assessment task for class imbalance problem, Nonlinear Analysis: Real World Applications 7 (4) (2006) 720–747.
[24] H. Ishibuchi, T. Nakashima, M. Nii, Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining, Springer-Verlag, 2004.
[25] H. Ishibuchi, T. Yamamoto, Rule weight specification in fuzzy rule-based classification systems, IEEE Transactions on Fuzzy Systems 13 (2005) 428–435.
[26] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intelligent Data Analysis Journal 6 (5) (2002) 429–450.
[27] T. Lane, C.E. Brodley, Approaches to online learning and concept drift for user identification in computer security, in: KDD, 1998.
[28] V. López, A. Fernández, J.G. Moreno-Torres, F. Herrera, Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics, Expert Systems with Applications 39 (7) (2012) 6585–6608.
[29] J.G. Moreno-Torres, F. Herrera, A preliminary study on overlapping and data fracture in imbalanced domains by means of genetic programming-based feature extraction, in: Proceedings of the 10th International Conference on Intelligent Systems Design and Applications (ISDA'10), 2010.
[30] J.G. Moreno-Torres, T. Raeder, R. Aláiz-Rodríguez, N.V. Chawla, F. Herrera, A unifying view on dataset shift in classification, Pattern Recognition 45 (1) (2012) 521–530.
[31] J.G. Moreno-Torres, J.A. Sáez, F. Herrera, Study on the impact of partition-induced dataset shift on k-fold cross-validation, IEEE Transactions on Neural Networks and Learning Systems 23 (8) (2012) 1304–1313.
[32] A. Orriols-Puig, E. Bernadó-Mansilla, Evolutionary rule-based systems for imbalanced datasets, Soft Computing 13 (3) (2009) 213–225.
[33] A. Orriols-Puig, E. Bernadó-Mansilla, D.E. Goldberg, K. Sastry, P.L. Lanzi, Facetwise analysis of XCS for problems with class imbalances, IEEE Transactions on Evolutionary Computation 13 (2009) 260–283.
[34] J. Quinlan, C4.5: Programs for Machine Learning, Morgan Kauffman, 1993.
[35] D. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman & Hall/CRC, 2006.
[36] H. Shimodaira, Improving predictive inference under covariate shift by weighting the log-likelihood function, Journal of Statistical Planning and Inference 90 (2) (2000) 227–244.
[37] A. Storkey, When training and test sets are different: characterizing learning transfer, in: J.Q. Candela, M. Sugiyama, A. Schwaighofer, N.D. Lawrence (Eds.), Dataset Shift in Machine Learning, MIT Press, 2009, pp. 3–28.
[38] Y. Sun, A.K.C. Wong, M.S. Kamel, Classification of imbalanced data: a review, International Journal of Pattern Recognition and Artificial Intelligence 23 (4) (2009) 687–719.
[39] Y. Tang, Y.-Q. Zhang, N.V. Chawla, S. Kresser, SVMs modeling for highly imbalanced classification, IEEE Transactions on Systems, Man and Cybernetics, Part B 39 (1) (2009) 281–288.
[40] K.M. Ting, An instance-weighting method to induce cost-sensitive trees, IEEE Transactions on Knowledge and Data Engineering 14 (3) (2002) 659–665.
[41] M. Wasikowski, X.-W. Chen, Combating the small sample class imbalance problem using feature selection, IEEE Transactions on Knowledge and Data Engineering 22 (10) (2010) 1388–1400.
[42] G.M. Weiss, Mining with rarity: a unifying framework, SIGKDD Explorations 6 (1) (2004) 7–19.
[43] G.M. Weiss, F.J. Provost, Learning when training data are costly: the effect of class distribution on tree induction, Journal of Artificial Intelligence Research 19 (2003) 315–354.

[44] G.M. Weiss, Y. Tian, Maximizing classifier utility when there are data acquisition and modeling costs, Data Mining and Knowledge Discovery 17 (2) (2008) 253–282.
[45] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, Machine Learning 23 (1) (1996) 69–101.
[46] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, second ed., Morgan Kaufmann, San Mateo, CA, 2005.
[47] K. Yamazaki, M. Kawanabe, S. Watanabe, M. Sugiyama, K.-R. Müller, Asymptotic bayesian generalization error when training and test distributions are different, in: Z. Ghahramani (Ed.), ICML, ACM International Conference Proceeding Series, vol. 227, ACM, 2007.
[48] Q. Yang, X. Wu, 10 Challenging problems in data mining research, International Journal of Information Technology and Decision Making 5 (4) (2006) 597–604.
[49] B. Zadrozny, C. Elkan, Learning and making decisions when costs and probabilities are both unknown, in: Proceedings of the 7th International Conference on Knowledge Discovery and Data Mining (KDD'01), 2001.
[50] B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting, in: Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03), 2003.
[51] J.H. Zar, Biostatistical Analysis, Prentice Hall, Upper Saddle River, New Jersey, 1999.
[52] X. Zeng, T.R. Martinez, Distribution-balanced stratified cross validation for accuracy estimation, Journal of Experimental and Theoretical Artificial Intelligence 12 (1) (2000) 1–12.

3. A study on the Scalability of FRBCSs for Imbalanced Datasets in the Big Data Scenario

The journal paper associated to this part is:

3.1. Cost-Sensitive Linguistic Fuzzy Rule-Based Classification Systems under the MapReduce Framework for Imbalanced Big Data

- V. López, S. del Río, J. M. Benítez, F. Herrera, Cost-Sensitive Linguistic Fuzzy Rule Based Classification Systems under the MapReduce Framework for Imbalanced Big Data. Fuzzy Sets and Systems, doi: 10.1016/j.fss.2014.01.015, ***in press (2014)***

  - Status: **Published (in press)**.
  - Impact Factor (JCR 2012): 1.749.
  - Subject Category: Computer Science, Theory & Methods. Ranking 17 / 100 (**Q1**).
  - Subject Category: Mathematics, Applied. Ranking 21 / 247 (**Q1**).
  - Subject Category: Statistics & Probability. Ranking 17 / 117 (**Q1**).

Available online at www.sciencedirect.com

ScienceDirect

Fuzzy Sets and Systems ••• (••••) •••–•••

FUZZY sets and systems

www.elsevier.com/locate/fss

Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data

Victoria López *, Sara del Río, José Manuel Benítez, Francisco Herrera

Dept. of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, Granada, Spain

Abstract

Classification with big data has become one of the latest trends when talking about learning from the available information. The data growth in the last years has rocketed the interest in effectively acquiring knowledge to analyze and predict trends. The variety and veracity that are related to big data introduce a degree of uncertainty that has to be handled in addition to the volume and velocity requirements. This data usually also presents what is known as the problem of classification with imbalanced datasets, a class distribution where the most important concepts to be learned are presented by a negligible number of examples in relation to the number of examples from the other classes. In order to adequately deal with imbalanced big data we propose the Chi-FRBCS-BigDataCS algorithm, a fuzzy rule based classification system that is able to deal with the uncertainly that is introduced in large volumes of data without disregarding the learning in the underrepresented class. The method uses the MapReduce framework to distribute the computational operations of the fuzzy model while it includes cost-sensitive learning techniques in its design to address the imbalance that is present in the data. The good performance of this approach is supported by the experimental analysis that is carried out over twenty-four imbalanced big data cases of study. The results obtained show that the proposal is able to handle these problems obtaining competitive results both in the classification performance of the model and the time needed for the computation.

Keywords: Fuzzy rule based classification systems; Big data; MapReduce; Hadoop; Imbalanced datasets; Cost-sensitive learning

1. Introduction

The development and maturity of the information technologies has enabled an exponential growth on the data that is produced, processed, stored, shared, analyzed and visualized. According to IBM [1], in 2012, every day 1.5 quintillion bytes of data are created, which means that the 90% of the data created in the world has been produced in the last two years. Big data [2] encompass a collection of datasets whose size and complexity challenges the standard database management systems and defies the application of knowledge extraction techniques. This data

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: vlopez@decsai.ugr.es (V. López), srio@decsai.ugr.es (S. del Río), J.M.Benitez@decsai.ugr.es (J.M. Benítez), herrera@decsai.ugr.es (F. Herrera).

http://dx.doi.org/10.1016/j.fss.2014.01.015
0165-0114/

comes from a wide range of sources such as sensors, digital pictures and videos, purchase transactions, social media posts, everywhere [3].

This generation and collection of large datasets has further encouraged the analysis and knowledge extraction process with the belief that with more data available, the information that could be derived from it will be more precise. However, the standard algorithms that are used in data mining are not usually able to deal with these huge datasets [4]. In this manner, classification algorithms must be redesigned and adapted considering the solutions that are being used in big data so that they are able to be used under these premises maintaining its predictive capacity.

One of the complications that make difficult the extraction of useful information from datasets is the problem of classification with imbalanced data [5,6]. This problem occurs when the number of instances of one class (positive or minority class) is substantially smaller than the number of instances that belong to the other classes (negative or majority classes). The importance of this problem resides on its prevalence in numerous real-world applications such as telecommunications, finances, medical diagnosis and so on. In this situation, the interest of the learning is focused towards the minority class as it is the class that needs to be correctly identified in these problems [7]. Big data is also affected by this uneven class distribution.

Standard classification algorithms do not usually work appropriately when dealing with imbalanced datasets. The usage of global performance measures for the construction of the model and the search for the maximum generalization capacity induce in algorithms a mechanism that tends to neglect the rules associated with instances of the minority class.

Fuzzy Rule Based Classification Systems (FRBCSs) [8] are effective and accepted tools for pattern recognition and classification. They are able to obtain a good precision while supplying an interpretable model for the end user through the usage of linguistic labels. Furthermore, the FRBCSs can manage uncertainty, ambiguity or vagueness in a very effective way. This trait is especially interesting when dealing with big data, as uncertainty is inherent to this situation. However, when dealing with big data, the information at disposal usually contains a high number of instances and/or features. In this scenario the inductive learning capacity of FRBCSs is affected by the exponential growth of the search space. This growth complicates the learning process and it can lead to scalability problems or complexity problems generating a rule set that is not interpretable [9].

To overcome this situation there have been several approaches that aim to build parallel fuzzy systems [10]. These approaches can distribute the creation of the rule base [11] or the post-processing of the built model, using a parallelization to perform a rule selection [12] or a lateral tuning of the fuzzy labels [13]. Moreover, a fuzzy learning model can be completely redesigned to obtain a parallel approach that decreases the computation time needed [14]. However, these models aim to reduce the wait for a final classification without damaging the performance and are not designed to handle huge volumes of data. In this manner, it is necessary to redesign the FRBCSs accordingly to be able to provide an accurate classification in a small lapse of time from big data.

Numerous solutions have been proposed to deal with imbalanced datasets [7,15]. These solutions are typically organized in two groups: data-level solutions [16,17], which modify the original training set to obtain a more or less balanced class distribution that can be used with any classifier, and algorithm-level solutions, which alter the operations of an algorithm so that the minority class instances have more relevance and are correctly classified. Cost-sensitive solutions [18,19] integrate both approaches as they are focused in reducing the misclassification costs, higher for the instances of the minority class.

The approaches used to tackle big data usually involve some kind of parallelization to efficiently process and analyze all the available data. One of the most popular frameworks for big data, MapReduce [20], organizes the processing in two key operations: a map process that is responsible for dividing the original dataset and processing each chunk of information, and a reduce process that collects the results provided in the previous step and combines those results accordingly including new treatment if necessary. This approach that divides the original dataset in parts can have a strong pernicious effect when dealing with imbalanced datasets as the data intrinsic characteristics impact is amplified. Specifically, the small sample size [21] is induced when the original dataset is shared out and the dataset shift problem [22] may also be encouraged in the process. The addition of these problems reinforce the necessity of properly dealing with imbalanced datasets, not only for the original imbalance that is present in the data but also for the occasioned problems that arise when the partitions are created.

In this paper, we present a FRBCS that is capable of classifying imbalanced big data which has been denoted as Chi-FRBCS-BigDataCS. The method is based on the Chi et al.'s approach [23], a classical FRBCS learning method, which has been modified to deal with imbalanced datasets and big data at the same time. The usage of a FRBCS

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

enables the treatment of the uncertainty that is inherent to real-world problems and especially, in big data problems, as the variety and veracity of the collected information pose a serious source of uncertainty and vagueness in the data. Fuzzy rules have demonstrated to adequately manage the uncertainty in a reasonable manner and therefore, FRBCSs seem to be a sensible choice to overcome this situation. Furthermore, FRBCSs [24,25], and specifically the Chi et al.'s method [26,27], have also been successfully applied to imbalanced domains where they do not only combat the problem of an uneven class distribution but they also face up to the challenge of the uncertainty in the class frontiers which comes up because of the borderline samples [28], the noise in the data [29] and the small disjuncts [30] among others.

Furthermore, using the Chi et al.'s method helps the classification in big data as it is a model that shows some characteristics that make it especially suitable to build a parallel approach instead of using a more state-of-the-art FRBCS method. The Chi et al.'s method is a simple approach that does not have complex operations and strong interactions between parts of the algorithm. This behavior allows a division of the processing operations without deeply degrading the performance of the algorithm. Moreover, all the rules generated by the Chi et al.'s method have the same structure: rules with as many antecedents as attributes in the dataset that only use one fuzzy label. Maintaining a common structure for the rules enormously benefits the combination and aggregation of rules that were created in different parallel operations and it greatly reduces the processing time. Other state-of-the-art methods may create more accurate rule bases, however, the associated rules do not have a common design and then, grouping them together substantially complicates the learning.

To deal with imbalanced big data, the proposed Chi-FRBCS-BigDataCS algorithm modifies the basic FRBCS approach combining two approaches:

- To deal with big data, the FRBCS method has been adapted following the MapReduce principles that direct a distribution of the work on several processing units.
- To address the imbalance that is present in the data, some modifications induced by cost-sensitive learning have been applied to the model. The use of a cost-sensitive approach is appropriate in this case as it does not introduce intensive computation operations and not adding thus extra runtime to the final model. For this, we propose a new rule weight computation, the Penalized Cost-Sensitive Certainty Factor (PCF-CS), an approach based on the original Penalized Certainty Factor that takes into consideration the misclassification costs.

In order to assess the performance of the suggested approach, we have used twenty-four imbalanced big data cases of study that provide information about how the proposal works, its strengths and its limitations. The experimental study is organized to analyze the performance related to two types of measures: an evaluation on the classification performance, which is measured by a well-known metric in imbalanced classification, the Area Under the ROC Curve [31], and an examination on the runtime of the approaches tested.

This paper is arranged as follows. In Section 2 some background information about classification with big data and imbalanced datasets is given. Next, Section 3 introduces some basic concepts about FRBCSs, describes the Chi et al.'s algorithm, and presents a scalability study to show the unfeasibility of this algorithm for big data. Section 4 shows how the basic Chi et al.'s algorithm is modified to address imbalanced datasets including the information about the new rule weight computation, and replays the scalability study to demonstrate that big data needs to be specifically addressed. Then, Section 5 characterizes the Chi-FRBCS-BigDataCS approach to deal with big data. Section 6 indicates the configuration of the experimental study, the results obtained and a discussion about them. Finally, the conclusions achieved in this work are shown in Section 7.

2. Classification with big data and imbalanced datasets

In this section we present some background information about the specific related data problems that we are trying to clarify. In Section 2.1 we provide information about big data, its characteristics and some solutions that have been proposed to overcome this challenge. Then, in Section 2.2, an overview about classification with imbalanced datasets is supplied featuring a description of its traits, given solutions, which are the main threats to properly solve this problem and how the performance of algorithms is measured in this scenario.

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

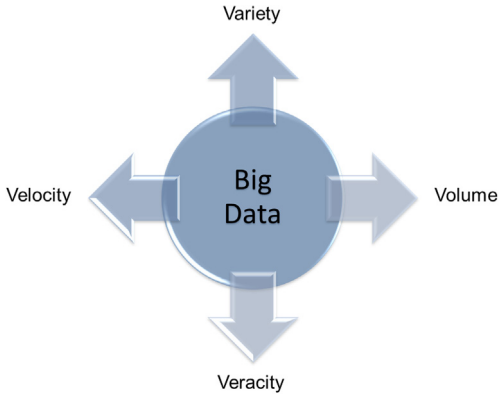


Fig. 1. The 4Vs of big data.

2.1. The difficulties of classification with big data

With the development of information technologies, organizations have had to face new challenges to analyze vast amounts of information. For this reason, the concept of "Big Data" is formulated, which is applied to all the information that cannot be processed or analyzed using traditional techniques or tools [32]. According to the definition given by the Gartner analyst Doug Laney in a 2001 MetaGroup research publication [33], we may describe big data as a 3Vs model (Volume, Velocity and Variety) [34,35]:

- **Volume:** It refers to the huge amount of data that needs to be processed, stored and analyzed.
- **Velocity:** It is an indication of how quickly the data needs to be analyzed so that it can provide an informed response.
- **Variety:** It is related to the different types of structured and unstructured data that organizations can accumulate such as tabular data (databases), hierarchical data, documents and e-mail, among others.

More recently, an additional V has been proposed by some organizations to describe the big data model [1] (Fig. 1): Veracity, which is an indication of data integrity and the trust on this information to make decisions. In this work we focus on effectively addressing the volume challenge, while trying to achieve reasonable results concerning the velocity model and also attempting to manage the uncertainty introduced by the variety and veracity.

These data volumes that we call big data are coming from different sources. For example, Facebook hosts approximately 10 billion photos, taking up one Petabyte of storage. The New York Stock Exchange generates about one Terabyte of new trade data per day, or the Internet Archive stores around 2 Petabytes of data, and is growing at a rate of 20 Terabytes per month [32].

Among the proposed solutions to the problem, one of the most popular approaches was proposed by Dean and Ghemawat, who worked at Google. They presented a parallel programming model, MapReduce, which is a framework for processing large volumes of data over a cluster of machines [20,36,37]. Generally, a MapReduce program contains two main phases: a map-function and a reduce-function. In the first phase, the input data is processed by the map-function, generating some intermediate results as the input of the reduce function in the second phase, which process the generated intermediate results to produce a final output.

Specifically, the MapReduce model is based on basic data structure which is the key-value pair, and all data processed in MapReduce is used in those key-value pair terms. In this manner, the map and reduce functions work as follows:

- **Map-function:** the master node performs a segmentation of the input dataset into independent blocks and distributes them to the worker nodes. Next, the worker node processes the smaller problem, and passes the answer back to its master node. In terms of key-value pairs, the map-function receives a key-value pair as input and emits a set of intermediate key-value pairs as output. Before the execution of a reduce function, the MapReduce library

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

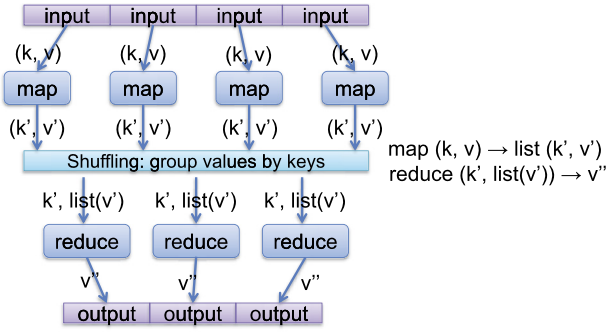


Fig. 2. The MapReduce programming model.

groups all intermediate values associated with the same intermediate key and transforms them to speed up the computation in the reduce function.

- **Reduce-function:** the master node collects the answers to all the sub-problems and combines them in some way to form the final output. Considering the key-value pairs, the reduce-function accepts the intermediate key provided by the MapReduce library and generates as final results the corresponding pair of key and value.

Fig. 2 depicts a typical MapReduce program with its map step and its reduce step. The terms k and v refer to the key and value pair respectively; k' and v' to the intermediate model and v'' to the output generated.

Apache Hadoop is the most popular implementation of the MapReduce programming model [32,38]. It is an open-source framework written in Java that supports the processing of large datasets in a distributed computing environment. Hadoop has a distributed file system, HDFS, that facilitates rapid data transfer rates among nodes and allows the system to continue operating uninterrupted in case of a node failure. The Apache Mahout project [39] is one of the most relevant tools that integrate machine learning algorithms in a Hadoop system.

However, following a MapReduce design is not always the best solution when dealing with big data [40]. Specifically, iterative algorithms are not able to obtain a good performance as they need to launch a MapReduce job for each iteration notably increasing the computation time due to the overhead. Therefore, there are some other open-source projects that are emerging to address big data as alternatives to MapReduce and Hadoop:

- **Spark [41]:** It is a cluster computing system that was developed in the UC Berkeley AMPLab and it is used to run large-scale applications such as spam filtering and traffic prediction. Spark provides primitives for in-memory cluster computing and APIs in Scala, Java and Python.
- **Apache Drill [42]:** It is a framework that supports data-intensive distributed applications for interactive analysis of large-scale datasets. Drill is a version of Google's Dremel system, which is a scalable, interactive ad-hoc query system for analysis of read-only nested data. Furthermore, its goal is to be able to scale to 10,000 servers or more and to be able to process Petabytes of data and trillions of records in seconds.

Some other incipient software projects are Twister [43], Ricardo [44], D3.js [45], HCatalog [46], Storm [47] or Impala [48], among others.

2.2. *Classification with imbalanced datasets*

Real-world classification problems typically present a class distribution where one or more classes have an insignificant number of examples in contrast with the number of examples from the other classes. This circumstance is known as the problem of classification with imbalanced datasets [5,6] and has been recognized as a challenge from the data mining community [49]. The main concern in this problem resides in the importance of the correct identification of the minority classes as they are the major focus of interest and their incorrect identification may entail high costs [18]. Imbalanced classification problems are found in diverse domains such as software defect prediction [50,51], finances [52], bioinformatics [53–55] and medical applications [56,57], just to mention some of them.

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 1
Confusion matrix for a two-class problem.

	Positive prediction	Negative prediction
Positive class	True Positive (TP)	False Negative (FN)
Negative class	False Positive (FP)	True Negative (TN)

Standard classification algorithms are usually unable to correctly deal with imbalanced datasets because they are built under the premise of obtaining the maximum generalization ability. In this manner, these algorithms try to obtain general rules that cover as many examples as possible, benefiting the majority class, while more specific rules that cover the minority class are discarded because of its small presence in the whole dataset. In this way, the minority class examples are treated like noise and therefore, these samples are finally neglected in the classification.

The imbalance ratio (IR) [58], which is the ratio of the number of instances in the majority class to the number of examples in the minority class $IR = \frac{\#maj\_class\_samples}{\#min\_class\_samples}$, is usually a clue to determine how difficult an imbalanced problem is. However, classification with imbalanced datasets is not only complicated by the dissimilar class distribution but also by some data intrinsic characteristics that interact with this issue aggravating the problem to a major extent than those difficulties in isolation [7]. Some of these data intrinsic characteristics include the presence of small disjuncts in the data [30], the small sample size for imbalanced classes [21], the overlapping between the classes [59], the presence of noisy [60] and borderline [61] examples and the dataset shift [22], which unites all the differences in the data distribution for the training and testing sets.

Big data techniques usually work in a parallel way dividing the original training set in subsets and distributing them along the processing units. This way of working is especially pernicious if the big data available is also imbalanced as it induces some of the aforementioned data problems: the small sample size problem and the dataset shift problem. In the first case, it is needed to establish a processing scheme that does not dramatically decrease the size of the new processed subsets. In the second case, a new subdivision of the dataset must be carefully done so that the subsets that are created for the training in each processing unit are as close as possible to the original training set. In this manner, we should avoid the prior probability shift [62], not changing the class distribution in the subsets, as well as the covariate shift [63], not changing the input attribute values distribution when the data portions are created.

Various approaches have been proposed to deal with imbalanced datasets [5–7,15]. These approaches are usually organized in two groups: *data-level* approaches and *algorithm-level* approaches. The data-level approaches [16,17] modify the original training set to obtain a more or less balanced distribution that is properly addressed by standard classification algorithms. This balancing process can be done adding examples to the minority class extending the dataset (over-sampling) or deleting examples from the majority class reducing the dataset (under-sampling). Algorithm-level approaches [25,64] adapt classification algorithms to guide the learning process towards the minority class. This adaptation can modify the inner way of working of an algorithm in favor of the minority class or it can even evidence the creation of new algorithms with this goal.

Additionally, cost-sensitive learning solutions include strategies at the data-level and the algorithm-level by considering variable misclassification costs for each class [19,65]. When dealing with imbalanced datasets it is more relevant to correctly classify minority instances than majority ones, and therefore, the cost associated to the misclassification of a minority instance should be higher than the cost associated to the contrary case: $Cost(min, maj) > Cost(maj, min)$. In this manner, cost-sensitive learning is either used as a direct approach that modifies how the algorithm works or is used as a meta-learning technique that modifies how the input or output information is processed [65,66]. Finally, another family of algorithms that has demonstrated a good behavior for imbalanced datasets is the ensembles of classifiers [67].

Selecting an appropriate performance measure is a vital decision when dealing with imbalanced datasets, not only to guide the construction of the model, but also to evaluate its achievement in comparison with other algorithms. The most used performance measure in classification, the overall classification accuracy, is not recommended when there is an uneven class distribution as it is biased towards the majority class: a classifier over a dataset with an IR of 9 that obtains a 90% of accuracy may not be a proper classifier as it may classify all the instances as belonging to the majority class, completely neglecting the minority class which is our interest in the problem.

In the imbalanced scenario, the evaluation of the classifiers performance should be computed considering specific metrics that observe the current class distribution. The confusion matrix (Table 1), which reports the results of correctly

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

or incorrectly classifying the examples of each class, leads to the obtaining of four metrics that describe both classes independently:

- **True positive rate** $TP_{rate} = \frac{TP}{TP+FN}$ is the percentage of positive instances correctly classified.
- **True negative rate** $TN_{rate} = \frac{TN}{FP+TN}$ is the percentage of negative instances correctly classified.
- **False positive rate** $FP_{rate} = \frac{FP}{FP+TN}$ is the percentage of negative instances misclassified.
- **False negative rate** $FN_{rate} = \frac{FN}{TP+FN}$ is the percentage of positive instances misclassified.

However, these measures are not satisfactory by themselves as we are seeking a good classification accuracy in both classes, and therefore, an approach to combine these measures is needed.

A graphical method that could be used to measure the performance of classification with imbalanced datasets is the Receiver Operating Characteristic (ROC) curve [68]. The ROC curve depicts the variation of the TP_{rate} against the FP_{rate} taking into account different decision threshold values. The Area Under the ROC Curve (AUC) metric [31] is able to provide a numerical performance measure that can be used to analyze the behavior of different learning algorithms. The AUC measure is computed obtaining the area of the ROC graphic. Specifically, we approximate this area following the next formula:

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \tag{1}$$

3. Classification with fuzzy rule based classification systems: The Chi et al.'s algorithm and the scalability problem

This section purpose is to provide the information needed to explain the necessity of modifying traditional methods when building FRBCSs in imbalanced big data. As a basis for the approach, we will recall some elementary definitions about FRBCSs in Section 3.1. Then, we will present the FRBCS that has been used to construct our approach, the Chi et al.'s algorithm in Section 3.2. Finally, we will show a scalability study in Section 3.3 that demonstrates the requirement of effectively addressing big data.

3.1. Fuzzy rule based classification systems

Among the diverse techniques that are used to deal with classification problems in data mining, FRBCSs are widely used because they produce an interpretable model with a reasonable prediction rate.

A FRBCS is formed of two main components: the knowledge base (KB) and the inference system. In a linguistic FRBCS, the KB is built from the rule base (RB) and the data base (DB). The RB contains all the rules that compose the model and the DB encodes the membership functions associated to the fuzzy data partitions that are related to the input attribute values. The inference system directs the way in which new examples are classified considering the information stored in the KB. The most advantageous situation arises when expert information is available, however, this is very unusual and automatic learning methods to build the KB are needed.

Let m be the number of training patterns $\mathbf{x}_p = (x_{p1}, \ldots, x_{pn})$ from C classes that form a classification problem, being x_{pi} is the ith attribute value ($i = 1, 2, \ldots, n$) of the p-th training pattern.

In this work, we use fuzzy rules of the following form to build our classifier:

$$\text{Rule } R_j: \text{ If } x_1 \text{ is } A_j^1 \text{ and } \ldots \text{ and } x_n \text{ is } A_j^n \text{ then Class} = C_j \text{ with } RW_j \tag{2}$$

where R_j is the label of the jth rule, $\mathbf{x} = (x_1, \ldots, x_n)$ is an n-dimensional pattern vector, A_j^i is an antecedent fuzzy set, C_j is a class label, and RW_j is the rule weight [69]. We use triangular membership functions as linguistic labels.

Numerous heuristics have been proposed to compute the rule weight [69]. A good choice for the computation of the rule weight is the Penalized Certainty Factor (PCF) [70], showed in Eq. (3):

$$RW_j = PCF_j = \frac{\sum_{x_p \in Class C_j} \mu_{A_j}(x_p) - \sum_{x_p \notin Class C_j} \mu_{A_j}(x_p)}{\sum_{p=1}^{m} \mu_{A_j}(x_p)} \tag{3}$$

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

where $\mu_{A_j}(x_p)$ is the matching degree of the pattern x_p with the antecedent part of the fuzzy rule R_j. We use the fuzzy reasoning method (FRM) of the wining rule [71], a classical approach, for the classification of new patterns by the RB. When a new pattern $\mathbf{x}_p = (x_{p1}, \ldots, x_{pn})$ needs to be classified, the winner rule R_w is decided as the rule verifying:

$$\mu_w(x_p) \cdot RW_w = \max\{\mu_j(x_p) \cdot RW_j;\ j = 1 \ldots L\} \tag{4}$$

The pattern $\mathbf{x}_p$ is classified as class C_w which is the class indicated in the consequent of the winner rule R_w. In the case where several rules obtain the same maximum value in Eq. (4) for the example $\mathbf{x}_p$ but with different classes on the consequent, the classification of the pattern $\mathbf{x}_p$ is rejected and therefore, no class is assigned to it. Similarly, if the example $\mathbf{x}_p$ does not match any rule in the RB, the classification is also rejected and no class is given to the example.

3.2. The Chi et al.'s algorithm for classification

As a base for our FRBCS for imbalanced big data, we have used a simple learning procedure to generate the KB. Specifically, we have considered the method described in [23], that we have called the Chi et al.'s rule generation method or Chi-FRBCS, which is an extension for classification problems of the well-known Wang and Mendel algorithm [72].

To build the KB, this FRBCS method tries to find the relationship between the variables of the problem and constitute an association between the domain of features and the domain of classes following the next steps:

1. *Establishment of the linguistic partitions:* Using the range of values for each attribute A_i, the linguistic fuzzy partitions that form the DB are computed with the same number of linguistic terms for all input variables, composed of symmetrical triangular-shaped and uniformly distributed membership functions.
2. *Generation of a fuzzy rule for each example* $\mathbf{x}_p = (x_{p1}, \ldots, x_{pn}, C_p)$*:* From each example present in the training set, a new fuzzy rule is created following the subsequent steps:
   (a) To compute the matching degree $\mu(\mathbf{x}_p)$ of the example with the different linguistic fuzzy labels for each attribute using a conjunction operator (represented with a T-norm operator).
   (b) To assign the example $\mathbf{x}_p$ to the different linguistic fuzzy labels that obtain the largest membership degree.
   (c) To generate a rule for the example $\mathbf{x}_p$. This rule will have as antecedent the linguistic fuzzy labels computed in the previous step and as consequent the class associated to the example C_p.
   (d) To compute the rule weight.

This procedure can generate several rules with the same antecedent. If the consequent of those rules belongs to the same class then, the replicated rules are deleted. However, if the consequent of those rules belongs to different classes then, only the rule with the highest weight is maintained in the RB.

3.3. Testing the scalability of the Chi-FRBCS algorithm

At this point, we want to test how the Chi-FRBCS algorithm is able to deal with huge amounts of data running a scalability test over the KDD Cup 1999 dataset from the UCI dataset repository [73]. The KDD Cup 1999 dataset features multiple classes while in our imbalanced scenario we are interested in problems with two classes. To test the Chi-FRBCS algorithm we have created several two-class big data cases of study derived from the KDD Cup 1999 dataset: specifically, the generated versions of the dataset use the *normal* and *DOS* connections as majority classes and the rest of attacks have been considered as minority classes. From these two-class datasets, we have created several imbalanced big data cases of study derived from it that differ in their size. From all the KDD Cup 1999 combinations we have selected three imbalanced big data cases of study that will be compared selecting only a percentage of samples from the original set maintaining the a priori probability between the classes. The percentage of the instances considered are the 10%, 25%, 40%, 50%, 60% and 75% and the experiments were run following a 5-fold stratified cross validation partitioning scheme. Further information about how the two-class sets are built can be found in Section 6.1.

Table 2 shows the information about the cases of study considered together with the average results in training and test for them. This table is divided by columns in four parts: the first three columns correspond to, for each case of

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 2
Average results for the Chi-FRBCS algorithm for the imbalanced big data cases of study using the AUC measure, number of rules and time elapsed.

Datasets	#Atts.	#Ex.	#Class(maj; min)	Chi-FRBCS				
				AUC_{tr}	AUC_{tst}	numRules	Runtime (s)	Runtime (hh:mm:ss.SSS)
kddcup_10_normal_versus_R2L	41	97 390	(97 278; 112)	0.5000	0.5000	131.6	1578.991	00:26:18.991
kddcup_25_normal_versus_R2L	41	243 476	(243 195; 281)	0.5036	0.5000	178.4	10 327.567	02:52:07.567
kddcup_40_normal_versus_R2L	41	389 562	(389 112; 450)	0.5047	0.5000	200.2	28 329.681	07:52:09.681
kddcup_50_normal_versus_R2L	41	486 953	(486 390; 563)	0.5062	0.5044	213.4	40 170.131	11:09:30.131
kddcup_60_normal_versus_R2L	41	584 343	(583 668; 675)	0.5046	0.5007	226.4	57 060.828	15:51:00.828
kddcup_75_normal_versus_R2L	41	730 429	(729 585; 844)	0.5067	0.5047	240.0	85 336.009	23:42:16.009
kddcup_full_normal_versus_R2L	41	973 907	(972 781; 1126)	0.5083	0.5030	219.2	174 285.276	48:24:45.276
kddcup_10_DOS_versus_R2L	41	388 449	(388 337; 112)	1.0000	0.9897	70.0	25 498.727	07:04:58.727
kddcup_25_DOS_versus_R2L	41	971 123	(970 842; 281)	0.9697	0.9645	79.0	141 280.704	39:14:40.704
kddcup_40_DOS_versus_R2L	41	1 553 798	(1 553 348; 450)	ND	ND	ND	ND	ND
kddcup_50_DOS_versus_R2L	41	1 942 248	(1 941 685; 563)	ND	ND	ND	ND	ND
kddcup_60_DOS_versus_R2L	41	2 330 697	(2 330 022; 675)	ND	ND	ND	ND	ND
kddcup_75_DOS_versus_R2L	41	2 913 371	(2 912 527; 844)	ND	ND	ND	ND	ND
kddcup_full_DOS_versus_R2L	41	3 884 496	(3 883 370; 1126)	ND	ND	ND	ND	ND
kddcup_10_DOS_versus_normal	41	485 615	(388 337; 97 278)	0.9973	0.9972	162.2	32 892.936	09:08:12.936
kddcup_25_DOS_versus_normal	41	1 214 037	(970 842; 243 195)	0.9973	0.9973	218.8	267 496.363	74:18:16.363
kddcup_40_DOS_versus_normal	41	1 942 460	(1 553 348; 389 112)	ND	ND	ND	ND	ND
kddcup_50_DOS_versus_normal	41	2 428 075	(1 941 685; 486 390)	ND	ND	ND	ND	ND
kddcup_60_DOS_versus_normal	41	2 913 690	(2 330 022; 583 668)	ND	ND	ND	ND	ND
kddcup_75_DOS_versus_normal	41	3 642 112	(2 912 527; 729 585)	ND	ND	ND	ND	ND
kddcup_full_DOS_versus_normal	41	4 856 151	(3 883 370; 972 781)	ND	ND	ND	ND	ND

study, the number of attributes (#Atts.), number of examples (#Ex.) and number of instances for each class (minority and majority). The fourth column is devoted to the results of the Chi-FRBCS algorithm. The results for that algorithm are organized in the following way: the first two columns correspond to the AUC average results in training and test, the third column shows the average number of rules created by the FRBCS and the fourth and fifth columns present the average response times in seconds and in the hh:mm:ss.SSS format. Please note, that the hh:mm:ss.SSS format stands for the hours, minutes, seconds and milliseconds spent in the computation. For each dataset we consider the average results of the partitions.

Analyzing the results we can observe the *ND* (Not Determinable) symbol, which indicates that the algorithm was not able to complete the experiment. The implementation tested has not been especially prepared for huge datasets and the appearance of the *ND* symbol means that the current algorithm cannot be scaled for big data, as it is not able to deal with datasets this size.

For example, for the dataset *kddcup_normal_versus_R2L*, the smallest one considered in this test, we can see that the algorithm was able to provide results for all the versions of the problem. The results in training and test do not provide huge differences between the different reduced versions while we are able to observe an increment in the number of rules and in the processing time as more data is available.

For the larger datasets, *kddcup_DOS_versus_R2L* and *kddcup_DOS_versus_normal*, we can observe that the reduced versions of the datasets which were not able to finish have considerably increased from the previous case as their size is more than four times the size of the *kddcup_normal_versus_R2L* dataset. Specifically, the Chi-FRBCS algorithm was not able to complete the experiment starting from the 40% reduced version of the *kddcup_DOS_versus_R2L* and the *kddcup_DOS_versus_normal* cases of study, and for the 25% versions, the elapsed time is huge in relation with the elapsed time for the 10% versions.

Furthermore, we could be tempted to address big data just reducing the size of the original training set so that the current model is able to provide a result; moreover, when the results obtained by the 10% reduced version provide a reasonable performance. However, the reduction in the dataset is not only performed in the training set but also in the test set which alters the conclusions we can extract. In [74], we can observe a set of experiments that are related to the training of a FRBCS with different versions of the same dataset reducing its size. Their findings showed that the performance in test (which was maintained) was truly affected by the usage of different training sets.

In this manner, we can conclude that the basic Chi-FRBCS is not an appropriate approach to address imbalanced big data and it is necessary to specifically address those problems to provide a FRBCS that is able to provide proper classification results in a sensible time.

4. The Chi et al.'s algorithm for classification with imbalanced datasets and the scalability problem

In this section we provide some knowledge about how the basic Chi-FRBCS model can be modified to be able to address imbalanced problems. First, in Section 4.1, we will present a proposal to improve the classification in this arduous scenario presenting an approach that uses a new rule weight computation based on the PCF. Then, in Section 4.2, we perform again a scalability study to show that the modifications introduced are adequate to deal with imbalanced data but they are not enough to effectively address imbalanced big data.

4.1. The Chi et al.'s algorithm for classification with imbalanced datasets: using the penalized cost-sensitive certainty factor

As stated in the previous section, we have selected as basis for our FRBCS for imbalanced big data the Chi-FRBCS method [23]. This procedure creates a KB that is able to perform reasonably well in a more or less balanced situation; however, the Chi-FRBCS does not perform properly when classifying imbalanced datasets [26]. To accurately deal with imbalanced datasets we need to modify the previous proposal using cost-sensitive learning so that it considers during the building of the model the different misclassification costs associated to the various examples. In this manner, the learning will be biased to better identify the instances of the minority class. This proposal will be called Chi-FRBCS-CS.

Chi-FRBCS-CS follows the same set of steps as Chi-FRBCS changing how the rule weights are computed. Specifically, using the PCF heuristic, we have included the misclassification costs in the rule weight developing the Penalized Cost-Sensitive Certainty Factor (PCF-CS). In this way, the PCF-CS is computed as:

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

$$\mathrm{RW}_j = \mathrm{PCF\text{-}CS}_j = \frac{\sum_{x_p \in ClassC_j} \mu_{A_j}(x_p) \cdot Cs(C_p) - \sum_{x_p \notin ClassC_j} \mu_{A_j}(x_p) \cdot Cs(C_p)}{\sum_{p=1}^{m} \mu_{A_j}(x_p) \cdot Cs(C_p)} \quad (5)$$

where $Cs(C_p)$ is the misclassification cost associated to class C_p, the class of the example $\mathbf{x}_p$.

The misclassification costs associated to any class should be given by the experts if knowledgeable information about the problem is available. Unfortunately, this situation is very rare and therefore, we need to establish a procedure to estimate these costs. In our approach we have selected the costs in the following way: $Cs(min, maj) = IR$ and $Cs(maj, min) = 1$. As requested in imbalanced datasets, the misclassification cost for the minority class is much higher than the misclassification cost associated to the majority class. Additionally, as the cost is dependent on the proportion between the majority and minority instances, this estimation is valid for datasets that range from a low imbalance level to extremely imbalanced datasets.

4.2. Testing the scalability of the Chi-FRBCS-CS algorithm

At this point, we want to reproduce the scalability test performed for the Chi-FRBCS-CS algorithm in order to test how the proposal works in imbalanced big data problems considering their size. In this manner, we use the same cases of study as in Section 3.3, the two-class variants of the KDD Cup 1999 dataset that were sampled at the 10%, 25%, 40%, 50%, 60% and 75% of its size. Table 3 shows the average results in training and test for the three selected imbalanced datasets for the Chi-FRBCS and Chi-FRBCS-CS algorithms. We include both algorithms to check the differences in behavior between them.

When comparing both approaches we can see that there are not many differences between both Chi-FRBCS versions and that the conclusions extracted for Chi-FRBCS can also be applied to Chi-FRBCS-CS. Specifically, we can recognize the presence of the *ND* symbol also for the Chi-FRBCS-CS algorithm and that it appears for the same cases of study where Chi-FRBCS has it. For instance, the *kddcup_normal_versus_R2L* dataset is processed in all cases while the larger datasets, *kddcup_DOS_versus_R2L* and *kddcup_DOS_versus_normal*, are only able to produce results when the smallest versions of the datasets are considered. In this manner, it can be inferred that the new approach for imbalanced datasets does not improve its behavior with respect to the dataset size.

When considering the AUC results in training and test, it can be detected a much better performance for the Chi-FRBCS-CS algorithm. This better results can be examined in the *kddcup_normal_versus_R2L* dataset where the AUC values experiment a greater improvement, going from a situation where the minority class is not properly identified to a situation where this minority instances are generally considered. This behavior can be seen in the different cases of study considered and does not depend on the data size. In the case of the *kddcup_DOS_versus_R2L* and *kddcup_DOS_versus_normal* datasets, the improvement is not as noticeable, however, the tendency to slightly improve the results is clear.

Viewing the number of rules generated by both approaches, the Chi-FRBCS-CS is the one that creates a model with the lesser number of rules. Regarding the time elapsed to complete the experiments, we can see that there is not a clear tendency between the two Chi-FRBCS versions. Even when they are able to provide results in the same cases, the time needed to finish the computation does not always benefit one algorithm over the other, which means that the calculation of the PCF-CS does not clearly increase the computation time needed while benefiting the classification performance.

Finally, we can conclude that the Chi-FRBCS-CS method is a step forward to deal with imbalanced datasets however, it is necessary to specifically address big data using techniques that have been designed to manage huge datasets, as standard learning algorithms have not been adapted to learn in this arduous situation.

5. The Chi-FRBCS algorithm for imbalanced big data: A MapReduce design

In this section, we will describe our proposal of a FRBCS for imbalanced big data, denoted as Chi-FRBCS-BigDataCS. This proposal is introduced in the following way: Section 5.1 presents a general overview of how the Chi-FRBCS algorithm is adapted for big data. Next, in Section 5.2, the building of the model is detailed. Later, Section 5.3 describes how the instances of a big dataset are classified considering the learned model. Finally, Section 5.4 presents a case of study over one of the imbalanced big data problems considered.

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 3
Average results for the sequential Chi-FRBCS and Chi-FRBCS-CS versions for the imbalanced big data cases of study using the AUC measure, number of rules and time elapsed.

Datasets	#Atts.	#Ex.	#Class(maj; min)	Chi-FRBCS					Chi-FRBCS-CS				
				AUC_{tr}	AUC_{tst}	numRules	Runtime (s)	Runtime (hh:mm:ss.SSS)	AUC_{tr}	AUC_{tst}	numRules	Runtime (s)	Runtime (hh:mm:ss.SSS)
kddcup_10_normal_versus_R2L	41	97 390	(97 278; 112)	0.5000	0.5000	131.6	**1578.991**	**00:26:18.991**	0.9729	**0.9499**	**119.0**	1599.831	00:26:39.831
kddcup_25_normal_versus_R2L	41	243 476	(243 195; 281)	0.5036	0.5000	178.4	10 327.567	02:52:07.567	0.9629	**0.9563**	**160.4**	**8426.257**	**02:20:26.257**
kddcup_40_normal_versus_R2L	41	389 562	(389 112; 450)	0.5047	0.5000	200.2	28 329.681	07:52:09.681	0.9637	**0.9587**	**180.4**	**21 274.452**	**05:54:34.452**
kddcup_50_normal_versus_R2L	41	486 953	(486 390; 563)	0.5062	0.5044	213.4	**40 170.131**	**11:09:30.131**	0.9649	**0.9625**	**195.0**	40 877.748	11:21:17.748
kddcup_60_normal_versus_R2L	41	584 343	(583 668; 675)	0.5046	0.5007	226.4	**57 060.828**	**15:51:00.828**	0.9634	**0.9597**	**205.6**	58 008.036	16:06:48.036
kddcup_75_normal_versus_R2L	41	730 429	(729 585; 844)	0.5067	0.5047	240.0	85 336.009	23:42:16.009	0.9657	**0.9638**	**218.8**	**84 191.977**	**23:23:11.977**
kddcup_full_normal_versus_R2L	41	973 907	(972 781; 1126)	0.5083	0.5030	219.2	**174 285.276**	**48:24:45.276**	0.9653	**0.9620**	**199.4**	176 795.885	49:06:35.885
kddcup_10_DOS_versus_R2L	41	388 449	(388 337; 112)	1.0000	**0.9897**	70.0	25 498.727	07:04:58.727	0.9999	0.9897	**64.6**	**25 448.700**	**07:04:08.700**
kddcup_25_DOS_versus_R2L	41	971 123	(970 842; 281)	0.9697	0.9645	79.0	141 280.704	39:14:40.704	0.9981	**0.9928**	**73.8**	**136 368.526**	**37:52:48.526**
kddcup_40_DOS_versus_R2L	41	1 553 798	(1 553 348; 450)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND
kddcup_50_DOS_versus_R2L	41	1 942 248	(1 941 685; 563)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND
kddcup_60_DOS_versus_R2L	41	2 330 697	(2 330 022; 675)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND
kddcup_75_DOS_versus_R2L	41	2 913 371	(2 912 527; 844)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND
kddcup_full_DOS_versus_R2L	41	3 884 496	(3 883 370; 1126)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND
kddcup_10_DOS_versus_normal	41	485 615	(388 337; 97 278)	0.9973	0.9972	162.2	**32 892.936**	**09:08:12.936**	0.9975	**0.9974**	**160.8**	33 670.214	09:21:10.214
kddcup_25_DOS_versus_normal	41	1 214 037	(970 842; 243 195)	0.9973	0.9973	218.8	**267 496.363**	**74:18:16.363**	0.9979	**0.9978**	**216.6**	273 740.590	76:02:20.590
kddcup_40_DOS_versus_normal	41	1 942 460	(1 553 348; 389 112)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND
kddcup_50_DOS_versus_normal	41	2 428 075	(1 941 685; 486 390)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND
kddcup_60_DOS_versus_normal	41	2 913 690	(2 330 022; 583 668)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND
kddcup_75_DOS_versus_normal	41	3 642 112	(2 912 527; 729 585)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND
kddcup_full_DOS_versus_normal	41	4 856 151	(3 883 370; 972 781)	ND	ND	ND	ND	ND	ND	ND	ND	ND	ND

5.1. General overview of the Chi-FRBCS algorithm for big data

The Chi-FRBCS-BigDataCS algorithm is an approach that can be used to classify imbalanced big data. It is a MapReduce design where each map process is responsible for building a RB using only the data included in its portion and where the reduce process is responsible for collecting and combining the RB generated by each mapper to form the final RB.

We will divide the description of the proposal in two parts: the first part is devoted to the description of the creation of the model, shown in Section 5.2, and the second part is dedicated to the explanation on how new instances are classified using the previous learned model, in Section 5.3. Both parts follow the MapReduce structure distributing all the computations needed along several processing units that manage different chunks of information, aggregating the results obtained in an appropriate manner.

In this description, we do not make a distinction between the steps that need to be followed to create a "normal" model that is able to process big data based on the Chi-FRBCS algorithm, Chi-FRBCS-BigData, and the steps needed to transform this model into our proposal, Chi-FRBCS-BigDataCS, based on the Chi-FRBCS-CS model. The differences in the computation of both models are related to the computation of the rule weight, as stated in Section 4.1, sharing most of the algorithm structure. In this manner, the transition to a big data model follows similar steps and only the variances associated to the cost-sensitive model will be stated when applicable.

The model presented is a FRBCS built on MapReduce using cost-sensitive learning for the following reasons:

- A FRBCS is able to deal with the uncertainty and imprecise information that emanates from big data, as those huge information sources become available from diverse sources that include a high variety while trying to cope with the veracity and trust on the data.
- The MapReduce framework is one of the most currently known alternatives to handle big data and has demonstrated that is capable to perform reasonably well in data mining problems producing even libraries like Mahout that include machine learning and data mining algorithms.
- In cost-sensitive learning, the addition of costs into the algorithm way of working does not heavily increase the time complexity while properly managing the imbalanced problem.

Finally, we have preferred the use of cost-sensitive learning instead of data preprocessing techniques to avoid an extra step in the building of the model following a MapReduce design. Over-sampling techniques would increase the size of the data to process, therefore increasing the computational needs, while under-sampling may disregard potentially useful examples which could be underestimated because of the subdivision induced by the MapReduce structure.

5.2. Building the knowledge base for the Chi-FRBCS-BigDataCS using a MapReduce design

In this section, we will describe how the KB is built from the original training set provided following a MapReduce procedure. This process is illustrated in Fig. 3 and consists of the following steps:

- *Initial:* In the CS version, the first step needs to estimate the costs for each class giving the minority class a greater cost than the majority class. This cost is estimated in the same way as described in Section 4.1, giving a misclassification cost of 1 for instances belonging to the majority class and a misclassification cost of IR for instances of the minority class.
  Next, in both versions of the algorithm, the domain of variation of each feature in the dataset is determined. Then, the different fuzzy membership labels that compose the DB are computed using these domains according to the number of labels considered.
  Finally, in order to comply with Hadoop way of working, the algorithm performs a segmentation of the training dataset into independent HDFS blocks. These blocks are then automatically replicated and transferred between the different cluster nodes thanks to the Hadoop environment that implements the MapReduce structure.
- *Map:* In this step each map task builds a RB with the data blocks of its data portion and generates a file containing the RB (called RB_i, see Fig. 3). More specifically, for each instance belonging to the mapper, a fuzzy rule is created in a similar way as described in Section 3.2: we first search for the linguistic fuzzy labels that match

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

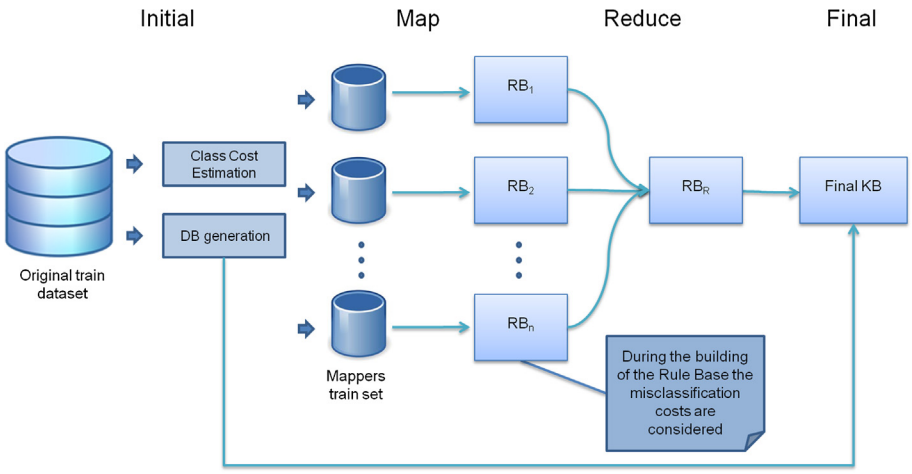


Fig. 3. A flowchart of how the building of the KB is organized in Chi-FRBCS-BigDataCS.

the attribute values of the current example, we select the among the matching fuzzy labels the ones that obtain the largest matching degree for each attribute, we build the rule using as antecedent the fuzzy labels previously selected and as consequent the class associated to the example and finally we compute the rule weight.
Please note that for computing the rule weight we use the PCF or PCF-CS for the Chi-FRBCS-BigData or Chi-FRBCS-BigDataCS methods, and that the set of examples used for the rule weight is the set of examples that belong to the current map process. In this manner, rules with the same antecedents and consequent can be generated by different mappers but they can have different rule weight values. Moreover, when a new rule is created in a mapper we check, as in the original Chi-FRBCS algorithm, if there is a rule with the same antecedents already in the mapper RB. In that case, if the consequent of the rule is also the same as the rule in the mapper RB, this rule is discarded while if the consequent of the new rule is different from the consequent of the previously created rule, then only the rule with the maximum weight is preserved.
In this manner, the Map step applies the original Chi-FRBCS classifier or the Chi-FRBCS-CS approach described in Section 4.1 to the data available in the data partition.

- *Reduce:* In this next step, the reduce process combines the RBs generated by each mapper (RB_i) to form the final RB (called RB_R, see Fig. 3). Specifically, the final RB is built from the RBs built from each mapper $RB_1, RB_2, \ldots, RB_n$ in a similar way as in the creation of new rules in each mapper (Fig. 3): we browse the rules that belong to the RB generated by each mapper, RB_i; if there is a rule in the final RB, RB_R, with the same antecedent as the rule we are trying to add we only maintain in the final RB, RB_R, the rule with the highest rule weight. In this case it is not necessary to check if the consequent is the same or not as we are maintaining the most powerful rules. Equivalent rules (rules with the same antecedent and consequent) can present different weights as they are computed in different mappers over different training sets.
  Please note that it is not needed to recompute the rules weights as we are selecting the most confident rules provided by each mapper. An alternative that would involve a new weight computation would have been the case where equivalent rules are combined to produce a new rule, for instance, computing an average weight between them. However, the direct selection of rules was preferred because of its simplicity which enables to speed up the algorithm in its reduce step.
- *Final:* In this last step, the RB that is generated in the reduce process (RB_R) and the DB that was calculated in the initial phase conform the KB that is provided as the output of the computation process. This output will be the entry data for the mechanism that classifies new examples.

Algorithms 1 and 2 show the pseudo-code of the Map function of the MapReduce job for the creation of the model phase. Algorithm 1 is devoted to obtaining all instances in a mapper's partition and the Hadoop framework calls it for each <key/value> pair in this partition. When the previous process is finished, Algorithm 2 is called for each mapper to build a RB with the data blocks of its data portion. Furthermore, Algorithm 3 gives the pseudo-code of the Reduce

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

function and is called when all mappers have finished, to combine the RBs generated by each mapper to form the final RB.

Algorithm 1 Map phase for the Chi-FRBCS-BigDataCS algorithm for the building of the model phase MAP(key, value):

Input: <key,value> pair, where key is the offset in bytes and value is the content of an instance.
Output: <key′,value′ > pair, where key′ is any Long value and value′ contains a RB.

```
1: instance ← INSTANCE_REPRESENTATION(value) {instances will contain all instances in this mapper's split}
2: instances ← instances.add(instance)
```

Algorithm 2 Map phase for the Chi-FRBCS-BigDataCS algorithm for the building of the model phase CLEANUP():

```
1: fuzzy_ChiBuilder.build(instances, posClass, posclassCost, negClassCost)
2: ruleBase ← fuzzy_ChiBuilder.getRuleBase()
3: EMIT (key, ruleBase)
```

Algorithm 3 Reduce phase for the Chi-FRBCS-BigDataCS algorithm REDUCE(key, values):

Input: <key,values> pair, where key is any Long value and values are the RBs generated by each mapper.
Output: <key′,value′ > pair, where key′ is a null value and value′ is the final RB.

```
1: while values.hasNext() do
2:     ruleBase ← values.getValue()
3:     for i = 0 to ruleBase.size() − 1 do
4:         if finalRuleBase.size() == 0 then
5:             finalRuleBase ← finalRuleBase.add(ruleBase.get(i));
6:         else
7:             if !finalRuleBase.duplicated(ruleBase.get(i)) then
8:                 finalRuleBase ← finalRuleBase.add(ruleBase.get(i));
9:             else
10:                if The consequent of those rules belongs to different classes then
11:                    rule ← finalRuleBase.getRuleWithHighestRuleWeight(ruleBase.get(i))
12:                    finalRuleBase ← finalRuleBase.add(rule);
13:                end if
14:            end if
15:        end if
16:    end for
17: end while
18: EMIT (null, finalRuleBase)
```

5.3. Classification of new patterns

In this section, we will describe how new instances belonging to a dataset are classified considering the KB built previously. When the MapReduce process devoted to the building of the KB has finished, a new MapReduce process is initiated to estimate the class associated to a dataset. Specifically, this phase is also based on a MapReduce design where each map process is responsible for estimating the class for the examples included in its data segment using the final KB previously generated. The process follows the next steps:

- *Initial:* In the same way as in the first step of the building of the model, this step performs a segmentation of the input dataset into independent HDFS blocks; replicates and transfers them to other machines to be finally processed independently by each map task at the same time. This step is automatically performed by the Hadoop system, the MapReduce implementation we are using.
- *Map:* In this next step, each map task estimates the class for the examples included in the data block available for the mapper using the FRM of the winner rule. In particular, for each example, we compute for all the rules in the RB the product of the rule weight with the compatibility degree between the linguistic fuzzy labels that

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

compose the antecedent of the rule and the example attribute values. The rule that obtains the highest value in this computation determines the new class for the example which is the class consequent of that rule.
- *Final:* In this last step, the predictions generated by each mapper are aggregated to conform the final predictions file. This step is just a concatenation of the results provided by each mapper without any extra computation.

It is important to note that the classification routine does not include a reduce step as it does not need to perform any kind of calculation to combine the results obtained by each mapper. Algorithm 4 gives the pseudo-code of the Map function of the MapReduce job for the classification phase. In this algorithm, Line (2) estimates the class for an instance and Line (5) saves the previously generated predictions.

Algorithm 4 Map phase for the Chi-FRBCS-BigDataCS algorithm for classifying phase MAP(key, value):

Input: <key,value> pair, where key is the offset in bytes and value is the content of an instance.
Output: <key′,value′ > pair, where key′ indicates the class of an instance and value′ contains its prediction.
1: *instance* ← *INSTANCE_REPRESENTATION*(*value*)
2: *prediction* ← *CLASSIFY*(*finalRuleBase*, *instance*)
3: *lkey* ← *lkey.set*(*instance.getClass*())
4: *lvalue* ← *lvalue.set*(*prediction*)
5: EMIT (lkey, lvalue)

5.4. Sample procedure of the Chi-FRBCS-BigDataCS algorithm for Imbalanced Big Data: A Case of Study

In order to illustrate how the Chi-FRBCS-BigDataCS algorithm works we have selected an imbalanced big data problem, the *kddcup_full_DOS_versus_U2R* dataset, to describe how the proposal behaves over it. This dataset is an imbalanced big data example with 41 input attributes and 3 883 422 instances. For this specific run, we have chosen the 5th partition of the 5-fcv used in the experimental study developed in this paper. This partition uses 3 105 769 instances for training (38 from the minority class, 3 105 731 from the majority class) and 777 653 for test (10 from the minority class, 777 643 from the majority class). We use 8 mappers in the Hadoop environment. Further information about this dataset is available in Section 6.1.

First, a MapReduce process is initiated in the building of the KB of the Chi-FRBCS-BigDataCS algorithm. The process follows the next steps:

- *Initial:* The first step is to estimate the costs for each class according to the procedure described in Section 4.1: the misclassification cost for instances in the majority class is 1 and the misclassification cost for examples that are associated to the minority class is the IR, that is, 81 729.76. The range of the different features of the dataset and the DB are also computed in this stage. Then, a segmentation of the training dataset into independent HDFS blocks is automatically performed; these blocks are replicated and transferred to other machines in the cluster and are processed by the map tasks in parallel. Each of these data blocks contains approximately 4.75 minority class samples and 388 216.38 majority class samples. Table 4 shows the actual number of instances from both classes available for each map task. This table shows that the distribution of samples is not completely stratified, as it is performed automatically by the Hadoop environment which does not consider the classes distribution.
- *Map:* Next, each map task builds a RB with the data available in its partition and generates a file containing the RB.
- *Reduce:* Later, the final RB is built from the RBs provided by each mapper, selecting from rules that share the same antecedent the rules with the greatest weight. In this manner, the reduce phase is able to decrement the number of final rules and easing the complexity of the model. Table 5 shows the number of rules by map task created in our case of study and the number of final rules. We have created 8 RBs, the number of map process that was made available in the Hadoop environment. We can observe that the number of rules has dramatically decreased from the 446 rules that were created by all the mappers to the 70 rules that finally compose the rule base.
- *Final:* Finally, the RB generated in the previous step and the DB calculated in the initial phase form the final KB that is provided as the output of the computation process.

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 4
Number of instances available for each map task for the Chi-FRBCS-BigDataCS version with 8 mappers.

kddcup_full_DOS_versus_U2R			
Mapper ID	Total instances	Minority class instances	Majority class instances
1	388 226	7	388 219
2	388 223	5	388 218
3	388 220	2	388 218
4	388 201	4	388 197
5	388 233	6	388 227
6	388 220	4	388 216
7	388 222	5	388 217
8	388 224	5	388 219

Table 5
Number of rules generated by map task and number of final rules for the Chi-FRBCS-BigDataCS version with 8 mappers.

kddcup_full_DOS_versus_U2R	
NumRules by mapper	Final numRules
RB_1 size: 60	RB_R size: 70
RB_2 size: 60	
RB_3 size: 55	
RB_4 size: 52	
RB_5 size: 49	
RB_6 size: 60	
RB_7 size: 52	
RB_8 size: 58	

Once we have finished the MapReduce process devoted to the building of the model, we generate a new MapReduce process to estimate the class for the examples of the training and test dataset:

- *Initial:* At the beginning, in the same way as in the building of the model, the algorithm performs a segmentation of the input dataset into independent HDFS blocks; replicates and transfers them to other machines to be finally processed independently by each map task concurrently.
- *Map:* Next, each map task estimates the classes of a subset of the dataset for every instance stored in it considering the final KB built previously, using the winning rule as FRM.
- *Final:* Finally, an aggregation of the predictions generated by each mapper compose the final predictions file.

6. Experimental study

In this section we show the experimental study carried out on the behavior of Chi-FRBCS-BigDataCS for imbalanced big data. First, in Section 6.1 we provide details of the classification problems chosen for the experimentation. Some of them have been used in previous sections for specific cases of study. Then, Section 6.2 introduces the algorithms selected for the comparison with the proposed approach and their configuration parameters. This section also details the infrastructure on which the experiments have been executed. Finally, Section 6.3 provides the performance results for the approaches using the AUC measure and shows the time elapsed for the datasets considered in the study.

6.1. Datasets used in the study

In order to analyze the quality of our approach, Chi-FRBCS-BigDataCS, we have run our experiments around three datasets from the UCI dataset repository [73]: the KDD Cup 1999 dataset, the Record Linkage Comparison Patterns (RLCP) dataset and the Poker Hand dataset. The KDD Cup 1999 dataset was used in the Third International Knowledge Discovery and Data Mining Tools Competition. It is a problem that represents a network intrusion detector,

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

and it aims to differentiate between *good* normal connections and *bad* connections that represent the different types of attacks. On the other hand, the underlying records in the Record Linkage Comparison Patterns Dataset stem from the epidemiological cancer registry of the German state of North Rhine-Westphalia. The Poker Hand dataset purpose is to predict poker hands.

Since the KDD Cup 1999 dataset and the Poker Hand dataset contain multiple classes, we have created several big data cases of study derived from them. More specifically, for the KDD Cup 1999 dataset we have generated new versions of the KDD Cup data using the *normal* and *DOS* connections as majority classes and the rest of attacks have been considered as minority classes. For the Poker Hand dataset we have obtained new versions using the 0 and 1 classes ("Nothing in hand" and "One pair" respectively) as majority classes and the rest of classes as minority classes. Moreover, we have also generated smaller versions of the original dataset selecting the 10% of the instances. For these reduced versions we have excluded the cases of study that contain less than 200 samples in their full versions, to make sure that in each mapper there is at least one sample of each class to learn the model.

Table 6 summarizes the data employed in this study and shows, for each dataset, the number of examples (#Ex.), number of attributes (#Atts.), class name of each class (minority and majority), number of instances for each class, class attribute distribution and IR.

In order to develop our study we use a 5-fold stratified cross validation partitioning scheme, that is, five random partitions of data with a 20% of the samples where the combination of 4 of them (80%) is considered as training set and the remaining one is treated as test set. For each dataset we consider the average results of the five partitions.

6.2. Algorithms and parameter settings

To verify the performance of the proposed model, we compare the results obtained by Chi-FRBCS-BigDataCS with the basic versions of the algorithm that solve the big data and imbalanced problems separately. Specifically, the algorithms considered in the study have been:

- **Chi-FRBCS** [23]: The classical fuzzy rule based classifier which was described in Section 3.2.
- **Chi-FRBCS-CS**: This is the proposed Chi-FRBCS version that introduces cost-sensitive learning modifying some of the Chi-FRBCS operations. This algorithm has been described in Section 4.1.
- **Chi-FRBCS-BigData**: This is the basic Chi-FRBCS version adapted to deal with big data. It is an algorithm that follows a MapReduce design which has been implemented under the Hadoop framework and is described in Section 5.
- **Chi-FRBCS-BigDataCS**: This is our final proposal, the modified version of the Chi-FRBCS-CS that has been prepared to take on imbalanced big data using a MapReduce scheme which has been implemented using Hadoop combined with cost-sensitive learning. This algorithm has also been described in Section 5.

The experiments associated to the sequential versions of the Chi-FRBCS algorithm have been run using the KEEL Software Tool [75,76].

Considering the parameters used in the experimentation, these algorithms use three fuzzy labels for each attribute, the product T-norm as conjunction operator in order to compute the matching degree of the antecedent of the rule with the example, PCF or PCF-CS (depending on the use of a CS version) to compute the rule weight and the FRM of the winning rule. Finally, only the approaches adapted for big data use a parameter related to the MapReduce procedure, which is the number of subsets of the original data that are created and provided for the map tasks. We have selected two different set of values for this parameter, as it has a direct impact on the AUC performance obtained and the runtime spent by the algorithms. Specifically, for the experiments on the reduced versions (10%) of the cases of study we have used 2, 4, 6, 8 and 16 mappers to have a better insight in the comparison with the sequential versions. For the full versions of the cases of study, we use 8, 16, 32 and 64 mappers to better address the big data cases under consideration. In this manner, the number of RBs created in the intermediate step of the algorithm depends on the number of map tasks.

With respect to the infrastructure used to perform the experiments, for the MapReduce designs, we have used the Atlas research group's cluster with 12 nodes, connected with a 1 Gb/s ethernet. Each node is composed by two Intel E5-2620 microprocessors (at 2 GHz, 15MB cache) and 64GB of main memory running under Linux CentOS 6.3. Furthermore, the cluster works with Hadoop 2.0.0 (Cloudera CDH4.3.0), where one node is configured as namenode

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 6
Summary of imbalanced datasets.

Datasets	#Ex.	#Atts.	Class (maj; min)	#Class(maj; min)	%Class(maj; min)	IR
kddcup_10_DOS_versus_normal	485 615	41	(DOS; normal)	(388 337; 97 278)	(79.968; 20.032)	3.99
kddcup_10_DOS_versus_PRB	392 447	41	(DOS; PRB)	(388 337; 4110)	(98.953; 1.047)	94.49
kddcup_10_DOS_versus_R2L	388 449	41	(DOS; R2L)	(388 337; 112)	(99.971; 0.029)	3467.29
kddcup_10_normal_versus_PRB	101 388	41	(normal; PRB)	(97 278; 4110)	(95.946; 4.054)	23.67
kddcup_10_normal_versus_R2L	97 390	41	(normal; R2L)	(97 278; 112)	(99.885; 0.115)	868.55
poker_10_0_vs_2	56 252	10	(0; 2)	(51 370; 4882)	(91.321; 8.679)	10.52
poker_10_0_vs_3	53 533	10	(0; 3)	(51 370; 2163)	(95.96; 4.04)	23.75
poker_10_0_vs_4	51 767	10	(0; 4)	(51 370; 397)	(99.233; 0.767)	129.40
poker_10_0_vs_5	51 575	10	(0; 5)	(51 370; 205)	(99.603; 0.397)	250.59
poker_10_0_vs_6	51 516	10	(0; 6)	(51 370; 146)	(99.717; 0.283)	351.85
poker_10_0_vs_7	51 393	10	(0; 7)	(51 370; 23)	(99.955; 0.045)	2233.48
poker_10_1_vs_2	48 191	10	(1; 2)	(43 309; 4882)	(89.869; 10.131)	8.87
poker_10_1_vs_3	45 472	10	(1; 3)	(43 309; 2163)	(95.243; 4.757)	20.02
poker_10_1_vs_4	43 706	10	(1; 4)	(43 309; 397)	(99.092; 0.908)	109.09
poker_10_1_vs_5	43 514	10	(1; 5)	(43 309; 205)	(99.529; 0.471)	211.26
poker_10_1_vs_6	43 455	10	(1; 6)	(43 309; 146)	(99.664; 0.336)	296.64
poker_10_1_vs_7	43 332	10	(1; 7)	(43 309; 23)	(99.947; 0.053)	1883.00
RLCP_10	574 913	2	(FALSE; TRUE)	(572 820; 2093)	(99.636; 0.364)	273.68
kddcup_DOS_versus_normal	4 856 151	41	(DOS; normal)	(3 883 370; 972 781)	(79.968; 20.032)	3.99
kddcup_DOS_versus_PRB	3 924 472	41	(DOS; PRB)	(3 883 370; 41 102)	(98.953; 1.047)	94.48
kddcup_DOS_versus_R2L	3 884 496	41	(DOS; R2L)	(3 883 370; 1126)	(99.971; 0.029)	3448.82
kddcup_DOS_versus_U2R	3 883 422	41	(DOS; U2R)	(3 883 370; 52)	(99.999; 0.001)	74 680.19
kddcup_normal_versus_PRB	1 013 883	41	(normal; PRB)	(972 781; 41 102)	(95.946; 4.054)	23.67
kddcup_normal_versus_R2L	973 907	41	(normal; R2L)	(972 781; 1126)	(99.884; 0.116)	863.93
kddcup_normal_versus_U2R	972 833	41	(normal; U2R)	(972 781; 52)	(99.995; 0.005)	18 707.33
poker_0_vs_2	562 530	10	(0; 2)	(513 702; 48 828)	(91.32; 8.68)	10.52
poker_0_vs_3	535 336	10	(0; 3)	(513 702; 21 634)	(95.959; 4.041)	23.75
poker_0_vs_4	517 680	10	(0; 4)	(513 702; 3978)	(99.232; 0.768)	129.14
poker_0_vs_5	515 752	10	(0; 5)	(513 702; 2050)	(99.603; 0.397)	250.59
poker_0_vs_6	515 162	10	(0; 6)	(513 702; 1460)	(99.717; 0.283)	351.85
poker_0_vs_7	513 938	10	(0; 7)	(513 702; 236)	(99.954; 0.046)	2176.70
poker_0_vs_8	513 719	10	(0; 8)	(513 702; 17)	(99.997; 0.003)	30 217.76
poker_0_vs_9	513 710	10	(0; 9)	(513 702; 8)	(99.998; 0.002)	64 212.75
poker_1_vs_2	481 925	10	(1; 2)	(433 097; 48 828)	(89.868; 10.132)	8.87
poker_1_vs_3	454 731	10	(1; 3)	(433 097; 21 634)	(95.242; 4.758)	20.02
poker_1_vs_4	437 075	10	(1; 4)	(433 097; 3978)	(99.09; 0.91)	108.87
poker_1_vs_5	435 147	10	(1; 5)	(433 097; 2050)	(99.529; 0.471)	211.27
poker_1_vs_6	434 557	10	(1; 6)	(433 097; 1460)	(99.664; 0.336)	296.64
poker_1_vs_7	433 333	10	(1; 7)	(433 097; 236)	(99.946; 0.054)	1835.16
poker_1_vs_8	433 114	10	(1; 8)	(433 097; 17)	(99.996; 0.004)	25 476.29
poker_1_vs_9	433 105	10	(1; 9)	(433 097; 8)	(99.998; 0.002)	54 137.13
RLCP	5 749 132	2	(FALSE; TRUE)	(5 728 201; 20 931)	(99.636; 0.364)	273.67

and jobtracker, and the rest are datanodes and task-trackers. For the sequential experiments we have used a cluster with Intel Core i7 930 microprocessors (at 2.8 GHz, 15MB cache) and 24GB of main memory connected with a 1 Gb/s ethernet. We acknowledge that the runtime comparisons between the sequential versions and the MapReduce designs are not performed in identical machines, however, the time advantage is obtained for the sequential versions which are, even in this case, notably slower than the Hadoop implementations.

6.3. Analysis of the Chi-FRBCS-BigDataCS behavior

In this part of the study, we want to analyze the behavior of the Chi-FRBCS-BigDataCS proposal in the scenario of imbalanced big data in contrast with the other learning proposals. This section is divided into two parts: the first part

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

(Section 6.3.1) is devoted to the presentation of the precision of our approach in terms of classification performance using the AUC measure; the second part (Section 6.3.2) is devoted to the analysis on the runtime of the model.

6.3.1. Analysis on the precision of the model

In this section, we present a set of experiments to illustrate and demonstrate the behavior of Chi-FRBCS-BigDataCS. These experiments are organized in two phases: the first one compares the behavior of the different alternatives using the cases of study that contain the 10% of the instances of the original datasets while the second one compares the behavior of the approaches over the full datasets considered in the study. The experiments where organized in this way to be able to contrast the results of the big data versions in relation with the serial versions of the algorithm for the smaller datasets. Additionally, this organization also permits to check how the results change when instead of using a reduced version of the dataset the whole dataset is utilized.

In Tables 7 and 8 we present the average results in training and test for the reduced versions (10%) of the imbalanced big data cases of study for the Chi-FRBCS and Chi-FRBCS-CS versions respectively. These tables are divided by columns in two parts: the first part corresponds to the results of the sequential variant while the second part is related to the big data variants of the Chi-FRBCS and Chi-FRBCS-CS algorithms respectively. Furthermore, the results for the big data alternatives are divided by columns in five parts, which correspond to the number of mappers used: 2, 4, 6, 8 and 16 mappers for each case.

Looking at the results we can observe that the performance obtained, both in training and test, is higher in most of the cases of study for the Chi-FRBCS-CS alternatives, the sequential approach and the big data adaptation for any number of mappers configuration. This situation demonstrates the positive influence of the usage of cost-sensitive learning when dealing with imbalanced data as the classifier is able to provide appropriate solutions in an arduous environment. Additionally, we can observe that the model does not present a strong overfitting on the training set in relation with the test set, as we cannot find huge differences between the results provided for both sets. For instance, for the *kddcup_10_normal_versus_PRB* dataset using the Chi-FRBCS-BigDataCS with 8 mappers, an AUC of 0.9728 in training is obtained which is closely followed by an AUC in test of 0.9723. There are even cases where the test set obtains a better performance than the training set such as *kddcup_10_normal_versus_R2L* for Chi-FRBCS-BigDataCS using 8 mappers with an AUC in training of 0.8747 and an AUC in test of 0.8784. This situation is caused by the usage of the PCF or PCF-CS to compute the rule weight as these measures try to make rules as general as possible considering the current dataset.

Next, we compare the results considering the cases of study derived from all original training sets in relation with the number of mappers considered. For the KDD Cup 1999 cases of study we find that the behavior of the Chi-FRBCS and Chi-FRBCS-CS approaches is not steady in relation to the number of mappers considered in the experiments. For instance, for the Chi-FRBCS sequential version, the test results achieved are worse than the results for the Chi-FRBCS-BigData approach. In this case, increasing the number of mappers may also increase the AUC metric, however, when the number of mappers is too high this performance is decreased. The Chi-FRBCS-CS sequential variant, is able to provide better test results than the Chi-FRBCS-BigDataCS proposal. However, there is not a clear optimal configuration for the number of mappers used, as the results are not stable when increasing that number of mappers. Furthermore, the worse results are obtained for the highest number of mappers considered in the experiment. In contrast, the training results provide more sensible results, decreasing the performance in a reasonable manner when the number of mappers is enlarged.

In the case of the Poker Hand cases of study we first discover that the results obtained for this set of data are a bit poor, as the AUC measure is usually ranging from 0.5 to 0.6. Similarly to the KDD Cup 1999 dataset, the Chi-FRBCS approaches are presenting erratic results where the sequential version provides worse AUC values than the Chi-FRBCS-BigData alternative, which is also improving when larger values for the number of mappers are used. In the case of the Chi-FRBCS-CS variants, the performance obtained is clearly related both in training and test with the number of mappers considered: the best performance is achieved by the sequential Chi-FRBCS-CS algorithm while the performance drops when bigger number of mappers are used.

The RLCP dataset is not able to properly identify instances from both classes in the Chi-FRBCS approaches, as the results obtained for all the variants and the number of mappers considered is 0.5. When the Chi-FRBCS-CS alternatives are tested, the RLCP provides reasonable AUC results with almost no variance when the sequential version is contrasted with smaller values for the number of mappers. Larger values for the number of mappers need to be compared to find a slight drop in accuracy.

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 7
Average results for the Chi-FRBCS versions for the imbalanced big data cases of study using the AUC measure.

Datasets	Chi-FRBCS		Chi-FRBCS-BigData									
			2 mappers		4 mappers		6 mappers		8 mappers		16 mappers	
	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}
kddcup_10_DOS_versus_normal	0.9973	0.9972	0.9993	**0.9993**	0.9993	**0.9993**	0.9993	**0.9993**	0.9993	**0.9993**	0.9992	**0.9993**
kddcup_10_DOS_versus_PRB	0.8440	0.8430	0.9055	0.9055	0.9052	0.9059	0.9112	**0.9116**	0.9029	0.9009	0.9088	0.9105
kddcup_10_DOS_versus_R2L	1.0000	0.9897	0.9951	0.9954	0.9988	0.9954	0.9987	**1.0000**	0.9988	**1.0000**	0.9013	0.8651
kddcup_10_normal_versus_PRB	0.8608	0.8589	0.9364	0.9376	0.9286	0.9284	0.9304	0.9311	0.9337	0.9332	0.9376	**0.9381**
kddcup_10_normal_versus_R2L	0.5000	0.5000	0.5000	0.5000	0.5120	0.5032	0.5560	0.5234	0.5419	**0.5359**	0.5195	0.5111
Average (kddcup)	0.8404	0.8377	0.8673	0.8676	0.8688	0.8664	0.8791	0.8731	0.8753	**0.8739**	0.8533	0.8448
poker_10_0_vs_2	0.5753	0.5052	0.5917	0.5108	0.6143	0.5146	0.6343	0.5182	0.6493	0.5195	0.6791	**0.5244**
poker_10_0_vs_3	0.5955	0.5082	0.6204	0.5180	0.6443	0.5222	0.6600	0.5291	0.6725	0.5310	0.7018	**0.5381**
poker_10_0_vs_4	0.5114	0.4956	0.5185	0.4999	0.5336	0.4998	0.5575	0.4998	0.5704	0.4997	0.6112	**0.5020**
poker_10_0_vs_5	0.7662	0.7039	0.8053	0.7857	0.8110	0.7992	0.8138	**0.8002**	0.8143	**0.8002**	0.8258	0.8001
poker_10_0_vs_6	0.5928	0.4963	0.6128	0.4999	0.6321	**0.5044**	0.6454	**0.5044**	0.6659	**0.5044**	0.6972	0.5043
poker_10_0_vs_7	0.5748	0.4960	0.5902	**0.5000**	0.5891	**0.5000**	0.6044	**0.5000**	0.6044	**0.5000**	0.6595	**0.5000**
poker_10_1_vs_2	0.5558	0.4933	0.5749	0.5045	0.6027	0.5066	0.6183	0.5086	0.6330	0.5087	0.6667	**0.5111**
poker_10_1_vs_3	0.5503	0.4924	0.5756	0.5028	0.5991	0.5048	0.6134	0.5047	0.6288	0.5065	0.6502	**0.5082**
poker_10_1_vs_4	0.5022	0.4901	0.5205	**0.4999**	0.5398	0.4997	0.5419	0.4996	0.5550	0.4994	0.5862	0.4990
poker_10_1_vs_5	0.7040	0.6222	0.7171	0.6816	0.7331	**0.7049**	0.7365	0.6977	0.7332	0.7047	0.7434	0.7045
poker_10_1_vs_6	0.5545	0.4891	0.5750	**0.4999**	0.5986	0.4997	0.6037	0.4997	0.6107	0.4997	0.6388	0.4994
poker_10_1_vs_7	0.5831	0.4891	0.5831	**0.5000**	0.5792	**0.5000**	0.5992	**0.5000**	0.5750	**0.5000**	0.5950	**0.5000**
Average (poker)	0.5888	0.5235	0.6071	0.5419	0.6231	0.5463	0.6357	0.5468	0.6427	0.5478	0.6712	**0.5493**
RLCP_10	0.5000	**0.5000**	0.5000	**0.5000**	0.5000	**0.5000**	0.5000	**0.5000**	0.5000	**0.5000**	0.5000	**0.5000**
Total average	0.6538	0.6095	0.6734	0.6300	0.6845	0.6327	0.6958	0.6349	0.6994	**0.6357**	0.7123	0.6286

Table 8
Average results for the Chi-FRBCS cost-sensitive versions for the imbalanced big data cases of study using the AUC measure.

Datasets	Chi-FRBCS-CS		Chi-FRBCS-BigDataCS									
			2 mappers		4 mappers		6 mappers		8 mappers		16 mappers	
	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}
kddcup_10_DOS_versus_normal	0.9975	0.9974	0.9994	**0.9995**	0.9995	**0.9995**	0.9995	**0.9995**	0.9995	**0.9995**	0.9994	0.9993
kddcup_10_DOS_versus_PRB	0.9849	**0.9831**	0.9588	0.9578	0.9588	0.9575	0.9584	0.9573	0.9582	0.9569	0.9571	0.9569
kddcup_10_DOS_versus_R2L	0.9999	0.9897	0.9999	**0.9999**	0.9999	**0.9999**	0.9999	**0.9999**	0.9999	**0.9999**	0.9524	0.9318
kddcup_10_normal_versus_PRB	0.9707	0.9697	0.9733	**0.9730**	0.9728	0.9729	0.9733	0.9729	0.9728	0.9723	0.9687	0.9688
kddcup_10_normal_versus_R2L	0.9729	**0.9499**	0.9638	0.9161	0.9640	0.9216	0.8983	0.8909	0.8747	0.8784	0.7443	0.7428
Average (kddcup)	0.9852	**0.9780**	0.9790	0.9693	0.9790	0.9703	0.9659	0.9641	0.9610	0.9614	0.9244	0.9199
poker_10_0_vs_2	0.9075	0.5905	0.8847	**0.5911**	0.8476	0.5737	0.8315	0.5689	0.8164	0.5623	0.7865	0.5500
poker_10_0_vs_3	0.9536	0.6173	0.9119	**0.6213**	0.8652	0.5960	0.8358	0.5824	0.8148	0.5727	0.7845	0.5587
poker_10_0_vs_4	0.9899	**0.5787**	0.9523	0.5633	0.8504	0.5324	0.7800	0.5287	0.7642	0.5185	0.7224	0.5190
poker_10_0_vs_5	0.9921	**0.8756**	0.9793	0.8706	0.9238	0.8399	0.8685	0.8120	0.8554	0.8097	0.8311	0.7997
poker_10_0_vs_6	0.9977	0.5082	0.9309	**0.5148**	0.8344	0.5116	0.8165	0.5117	0.8128	0.5115	0.7955	0.5115
poker_10_0_vs_7	0.9990	0.4947	0.8666	0.4999	0.8506	0.4999	0.8245	0.4999	0.8084	0.4999	0.7936	**0.5000**
poker_10_1_vs_2	0.8818	0.5306	0.8580	**0.5481**	0.8198	0.5380	0.8016	0.5394	0.7848	0.5313	0.7563	0.5261
poker_10_1_vs_3	0.9338	0.5368	0.8874	**0.5423**	0.8206	0.5337	0.7885	0.5279	0.7664	0.5203	0.7218	0.5104
poker_10_1_vs_4	0.9800	0.5359	0.9135	**0.5402**	0.7787	0.5193	0.7219	0.5086	0.6848	0.5101	0.6459	0.5073
poker_10_1_vs_5	0.9918	**0.8782**	0.9649	0.8250	0.9101	0.7881	0.8394	0.7369	0.8144	0.7299	0.7608	0.7105
poker_10_1_vs_6	0.9939	0.4923	0.8518	0.4974	0.7488	0.4986	0.6951	0.4989	0.6940	0.4989	0.6819	**0.4991**
poker_10_1_vs_7	0.9981	0.4868	0.8867	0.4996	0.7085	**0.4999**	0.6880	**0.4999**	0.6111	**0.4999**	0.6111	**0.4999**
Average (poker)	0.9683	**0.5938**	0.9073	0.5928	0.8299	0.5776	0.7909	0.5679	0.7690	0.5638	0.7410	0.5577
RLCP_10	0.9135	**0.9135**	0.9135	**0.9135**	0.9135	**0.9135**	0.9135	**0.9135**	0.9110	0.9104	0.9070	0.9069
Total average	0.9699	**0.7183**	0.9276	0.7152	0.8759	0.7053	0.8463	0.6972	0.8302	0.6935	0.8011	0.6777

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

In all these cases of study we can say that there is not a strong degradation in the performance when using the MapReduce versions. Specifically, the Chi-FRBCS-BigDataCS is more affected by the increasing number of mappers than Chi-FRBCS-BigData, however, this behavior is expected because increasing the number of portions induces the dataset shift problem and the small sample size, situations that have a pernicious effect when dealing with imbalanced datasets. To test the influence of the small sample size problem when different number of mappers are considered, we show in Table 9 the diverse number of minority and majority class instances by mapper for the Chi-FRBCS-BigData versions. Please note that the number of instances per mapper for Chi-FRBCS-BigData and Chi-FRBCS-BigDataCS is the same, because the initial stage in both algorithms is identical: the framework automatically divides the data in different information portions that are then copied and distributed to all the mapper processes considered.

As it is expected, the number of instances per mapper from each class is drastically reduced when higher values for the number of mappers are obtained. This decrement on the available number of instances is observed in both classes, however, it has a greater impact on the minority class. The minimum average number of samples per mapper in the most adverse situation for the majority class is 2164.75 for all the reduced versions considered, which is a reasonable number of samples to learn the associated fuzzy rules. However, when the number of minority class samples is observed for the maximum number of mappers considered, we find several cases of study that do not have at least 7 minority class samples per mapper. In these cases we encounter the small sample size problem which is responsible for the poor results achieved. The small sample size problem also influences the increasing drop in the performance of the algorithms when larger values for the number of mappers are utilized. For instance, the cases of study with the lesser number of minority class instances, like *poker_10_0_vs_7* and *poker_10_1_vs_7*, obtain very poor results being unable to properly identify instances from both classes. In the *kddcup_10_normal_versus_R2L* case of study we can also observe the dramatical drop in the performance, going from an AUC value of 0.9693 when 2 mappers are used to 0.7428 for 16 mappers, as we range from 45.60 minority class instances by mapper to 5.70.

Table 10 shows the average results in training and test for the full imbalanced big data cases of study. This table is divided by columns in two parts: the first column is related to the Chi-FRBCS-BigData algorithm while the second column is related with the cost-sensitive alternative, the Chi-FRBCS-BigDataCS algorithm. As in the preceding case, these algorithms organize their results by columns in four parts according to the number of mappers: 8, 16, 32 and 64 respectively. Please note that the sequential versions were not included in this table since these approaches were not able to complete an experiment with data this size as it was shown in the scalability studies (Sections 3.3 and 4.2).

On the first hand, we can observe a similar behavior between the reduced datasets in relation with the full datasets. Specifically, Chi-FRBCS-BigDataCS is able to provide a much better performance than Chi-FRBCS-BigData for all the diverse number of mappers tested. The differences between the training and test results are observed only for the Poker Hand cases of study which means that overfitting appears when the size of the training set is smaller.

On the other hand, the results related to the number of mappers used also resemble the results obtained for the sequential versions. For instance, when examining the number of mappers used for the big data developments, we can see that as the number of mappers increases and therefore the data available for each mapper is reduced, our proposal Chi-FRBCS-BigDataCS maintains a slight decrease in performance whereas the Chi-FRBCS-BigData alternative is not able to show a clear tendency.

When we take a closer look grouping together the cases of study that are derived from the same datasets we can observe that the general conclusions extracted can also be applied to these groups. Specifically, the KDD Cup 1999 cases of study follow this different behavior for Chi-FRBCS-BigData and Chi-FRBCS-BigDataCS. Chi-FRBCS-BigData does not show a clear trend for diverse values of the number of mappers, while the Chi-FRBCS-BigDataCS method decrements its performance when larger number of mappers are utilized.

The Poker Hand cases of study also closely follow this disposition, not having a shift according to the number of mappers for the Chi-FRBCS-BigData method but degrading the performance of the Chi-FRBCS-BigDataCS method for high values of the number of mappers considered. In addition, we also observe that the values obtained for the AUC measure are still poor for these cases of study, however, they are better than the results obtained for the reduced 10% cases of study previously analyzed.

The RLCP dataset shows a similar behavior to the one previously analyzed. The Chi-FRBCS-BigData approach does not show a correct classification of the samples considered as it obtains an AUC value of 0.5. For the Chi-FRBCS-BigDataCS, the results achieved, while being better, do not vary much with respect to the number of mappers. For the smaller values of the number of mappers the AUC results are the same, while they are slightly diminished when larger values are considered.

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 9
Average number of minority and majority class instances by mapper for the Chi-FRBCS-BigData and Chi-FRBCS-BigDataCS versions.

Datasets	Chi-FRBCS-BigData and Chi-FRBCS-BigDataCS									
	2 mappers		4 mappers		6 mappers		8 mappers		16 mappers	
	#min class	#maj class	#min class	#maj class	#min class	#maj class	#min class	#maj class	#min class	#maj class
kddcup_10_DOS_versus_normal	39 014.40	155 231.60	19 507.20	77 615.80	13 004.80	51 743.87	9753.60	38 807.90	4876.80	19 403.95
kddcup_10_DOS_versus_PRB	1612.00	155 366.80	806.00	77 683.40	537.33	51 788.93	403.00	38 841.70	201.50	19 420.85
kddcup_10_DOS_versus_R2L	40.80	155 338.80	20.40	77 669.40	13.60	51 779.60	10.20	38 834.70	5.10	19 417.35
kddcup_10_normal_versus_PRB	1639.20	38 916.00	819.60	19 458.00	546.40	12 972.00	409.80	9729.00	204.90	4864.50
kddcup_10_normal_versus_R2L	45.60	38 910.40	22.80	19 455.20	15.20	12 970.13	11.40	9727.60	5.70	4863.80
poker_10_0_vs_2	1952.00	20 548.80	976.00	10 274.40	650.67	6849.60	488.00	5137.20	244.00	2568.60
poker_10_0_vs_3	906.40	20 506.80	453.20	10 253.40	302.13	6835.60	226.60	5126.70	113.30	2563.35
poker_10_0_vs_4	162.00	20 544.80	81.00	10 272.40	54.00	6848.26	40.50	5136.20	20.25	2568.10
poker_10_0_vs_5	88.00	20 542.00	44.00	10 271.00	29.33	6847.33	22.00	5135.50	11.00	2567.75
poker_10_0_vs_6	52.40	20 554.00	26.20	10 277.00	17.46	6851.33	13.10	5138.50	6.55	2569.25
poker_10_0_vs_7	6.80	20 550.40	3.40	10 275.20	2.26	6850.13	1.70	5137.60	0.85	2568.80
poker_10_1_vs_2	1927.60	17 348.80	963.80	8674.40	642.53	5782.93	481.90	4337.20	240.95	2168.60
poker_10_1_vs_3	856.80	17 332.00	428.40	8666.00	285.60	5777.33	214.20	4333.00	107.10	2166.50
poker_10_1_vs_4	156.00	17 326.40	78.00	8663.20	52.00	5775.46	39.00	4331.60	19.50	2165.80
poker_10_1_vs_5	87.60	17 318.00	43.80	8659.00	29.20	5772.66	21.90	4329.50	10.95	2164.75
poker_10_1_vs_6	56.80	17 325.20	28.40	8662.60	18.93	5775.06	14.20	4331.30	7.10	2165.65
poker_10_1_vs_7	8.80	17 324.00	4.40	8662.00	2.93	5774.66	2.20	4331.00	1.10	2165.50
RLCP_10	827.60	229 137.60	413.80	114 568.80	275.87	76 379.20	206.90	57 284.40	103.45	28 642.20

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 10
Average results for the big data Chi-FRBCS versions for the full imbalanced big data cases of study using the AUC measure.

Datasets	Chi-FRBCS-BigData								Chi-FRBCS-BigDataCS							
	8 mappers		16 mappers		32 mappers		64 mappers		8 mappers		16 mappers		32 mappers		64 mappers	
	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}	AUC_{tr}	AUC_{tst}
kddcup_DOS_versus_normal	0.9992	0.9992	0.9992	0.9992	0.9993	**0.9993**	0.9992	0.9992	0.9993	**0.9993**	0.9993	**0.9993**	0.9993	**0.9993**	0.9993	**0.9993**
kddcup_DOS_versus_PRB	0.8639	0.8636	0.8636	0.8633	0.8639	0.8639	0.8634	0.8633	0.9558	**0.9557**	0.9556	0.9556	0.9553	0.9553	0.9546	0.9545
kddcup_DOS_versus_R2L	0.9941	0.9913	0.9881	0.9886	0.9779	0.9769	0.9942	0.9918	0.9999	**0.9999**	0.9999	**0.9999**	0.9999	**0.9999**	0.9999	**0.9999**
kddcup_DOS_versus_U2R	0.8544	0.8464	0.8544	0.8464	0.8544	0.8464	0.8544	0.8464	0.9387	**0.9366**	0.8960	0.8880	0.8960	0.8880	0.8960	0.8880
kddcup_normal_versus_PRB	0.8936	0.8932	0.8701	0.8693	0.8784	0.8788	0.8693	0.8690	0.9681	**0.9681**	0.9670	0.9671	0.9683	0.9679	0.9663	0.9659
kddcup_normal_versus_R2L	0.5086	0.5050	0.5089	0.5055	0.5197	0.5178	0.5246	0.5223	0.9626	**0.9616**	0.9460	0.9446	0.9101	0.9119	0.8298	0.8229
kddcup_normal_versus_U2R	0.5397	**0.5000**	0.5454	**0.5000**	0.5454	**0.5000**	0.5454	**0.5000**	0.5518	**0.5000**	0.5575	**0.5000**	0.5575	**0.5000**	0.5575	**0.5000**
Average (kddcup)	0.8076	0.7998	0.8042	0.7960	0.8056	0.7976	0.8072	0.7989	0.9109	**0.9030**	0.9030	0.8935	0.8981	0.8889	0.8862	0.8758
poker_0_vs_2	0.5240	0.5146	0.5310	0.5188	0.5400	0.5237	0.5480	0.5271	0.7371	**0.6689**	0.7048	0.6420	0.6597	0.6054	0.6126	0.5700
poker_0_vs_3	0.5338	0.5189	0.5477	0.5271	0.5626	0.5367	0.5751	0.5432	0.7950	**0.7132**	0.7422	0.6686	0.6744	0.6163	0.6250	0.5769
poker_0_vs_4	0.5005	0.5000	0.5012	0.5000	0.5065	0.5005	0.5140	0.5009	0.8262	**0.6755**	0.6752	0.5961	0.5884	0.5345	0.5526	0.5132
poker_0_vs_5	0.7341	0.7298	0.7483	0.7479	0.7488	0.7486	0.7489	0.7486	0.9686	**0.9588**	0.8977	0.8859	0.7707	0.7673	0.7490	0.7485
poker_0_vs_6	0.5445	0.5194	0.5553	0.5218	0.5645	0.5264	0.5719	0.5280	0.6559	**0.5611**	0.6131	0.5410	0.5969	0.5371	0.5943	0.5355
poker_0_vs_7	0.5935	0.5115	0.6220	0.5139	0.6562	0.5228	0.6704	0.5299	0.6935	0.5294	0.6776	**0.5318**	0.6825	**0.5318**	0.6825	**0.5318**
poker_0_vs_8	0.5000	0.5000	0.6262	0.5000	0.7422	0.5000	0.8333	0.6750	0.8396	**0.7750**	0.8396	**0.7750**	0.8396	**0.7750**	0.8396	**0.7750**
poker_0_vs_9	0.7500	**0.5000**	0.7708	**0.5000**	0.7708	**0.5000**	0.7708	**0.5000**	0.7708	**0.5000**	0.7708	**0.5000**	0.7708	**0.5000**	0.7708	**0.5000**
poker_1_vs_2	0.5032	0.5004	0.5071	0.5015	0.5125	0.5027	0.5185	0.5045	0.6681	**0.5761**	0.6354	0.5589	0.5942	0.5363	0.5575	0.5208
poker_1_vs_3	0.5021	0.5002	0.5056	0.5010	0.5114	0.5032	0.5180	0.5054	0.7004	**0.6088**	0.6383	0.5649	0.5760	0.5326	0.5393	0.5139
poker_1_vs_4	0.5010	0.5000	0.5016	0.5000	0.5035	0.5000	0.5057	0.4999	0.7511	**0.6191**	0.6054	0.5593	0.5283	0.5062	0.5154	0.5005
poker_1_vs_5	0.7483	0.7481	0.7484	0.7483	0.7486	0.7483	0.7498	0.7490	0.9745	**0.9617**	0.9017	0.8911	0.7814	0.7769	0.7488	0.7486
poker_1_vs_6	0.5047	0.5000	0.5146	0.5010	0.5244	0.5023	0.5313	0.5034	0.5826	**0.5163**	0.5457	0.5053	0.5387	0.5043	0.5388	0.5044
poker_1_vs_7	0.5077	**0.5016**	0.5089	0.5000	0.5180	0.5000	0.5278	0.5000	0.5517	0.5000	0.5492	0.5000	0.5457	0.5000	0.5438	0.5000
poker_1_vs_8	0.5125	**0.5000**	0.6583	**0.5000**	0.7140	**0.5000**	0.7693	**0.5000**	0.7745	**0.5000**	0.7745	**0.5000**	0.7745	**0.5000**	0.7745	**0.5000**
poker_1_vs_9	0.8383	**0.6000**	0.8383	**0.6000**	0.8383	**0.6000**	0.8383	**0.6000**	0.8383	**0.6000**	0.8383	**0.6000**	0.8383	**0.6000**	0.8383	**0.6000**
Average (poker)	0.5811	0.5403	0.6053	0.5426	0.6226	0.5447	0.6369	0.5572	0.7580	**0.6415**	0.7131	0.6137	0.6725	0.5827	0.6552	0.5712
RLCP	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.9134	**0.9134**	0.9134	**0.9134**	0.9134	**0.9134**	0.9093	0.9092
Total average	0.6438	0.6143	0.6590	0.6147	0.6709	0.6166	0.6809	0.6253	0.8091	**0.7291**	0.7768	0.7078	0.7483	0.6858	0.7331	0.6741

The general tendency that incurs in a drop in the performance for good performing algorithms appears usually when a more parallel solution is compared with a less parallel solution or sequential solution, as only partial information is available for the computation in contrast with larger portions of information that can even cover the whole information available. However, this undesirable effect is not only related to the less quantity of data available but also to the induction of the small sample size problem that further hinders the classification performance in imbalanced situations, which is noticeable in Chi-FRBCS-BigDataCS. To measure the effect of this problem, we present in Table 11 the number of minority and majority class instances by mapper for the Chi-FRBCS-BigData versions. We would like to remind the reader that the number of instances per mapper for Chi-FRBCS-BigData and Chi-FRBCS-BigDataCS is the same, because the initial stage in both algorithms is identical.

This table displays the enormous reduction in the number of samples from each class when larger values for the number of mappers are utilized. In this case, as in the reduced versions, the decrement of the available samples from each class is noticeable, but the influence of the minority class is greater than the influence of the majority class. For the full datasets, in the most difficult scenario, the average number of majority class instances per mapper is 5413.60, which is clearly a fair amount of instances to build a model. However, when we turn to the minority class instances, in the worst case scenario we can find several cases of study that are not even able to provide 1 minority class per mapper, which are usually cases that are not able to properly identify both classes in the test set. Furthermore, when we look at not so dramatic cases of study, we can also find problems with 15 to 20 minority class samples. In these cases, even when there are more instances, the quantity of them is risible with respect to the number of majority class samples, which means that these cases also suffer from the small sample size problem. Furthermore, the small sample size problem aggravates the decrement in the performance for the larger values of the number of mappers. For instance, the *kddcup_normal_versus_R2L* dataset shows an AUC metric of 0.9616 when 8 mappers are used, while this value lowers to 0.8229 when the number of mappers is set to 64.

We acknowledge that this decrement in precision is inevitable when a division of the input data is needed to speed up the classification process; however, these results show that it is of the utmost importance to select an appropriate threshold to perform the data division for the processing, especially in the presence of imbalanced datasets. When a good threshold is established, the downfall in precision is admissible but when that threshold does not fit the problem considered, we can see a lethal reduction in the performance invalidating all the learning process followed due to the small sample size problem.

6.3.2. Analysis on the runtime of the model

Tables 12 and 13 show the time elapsed in seconds and in the hh:mm:ss.SSS format (hours, minutes, seconds, milliseconds) for the reduced versions (10%) of the imbalanced big data cases of study for the Chi-FRBCS and the Chi-FRBCS-BigData alternatives, and for the Chi-FRBCS-CS and Chi-FRBCS-BigDataCS methods respectively. These tables are divided by columns in two parts: the first part corresponds to the results of the sequential variant while the second part is related to the big data variants of the Chi-FRBCS and Chi-FRBCS-CS algorithms respectively. Moreover, the results for the big data versions are divided by columns in five parts which correspond to the number of mappers used: 2, 4, 6, 8 and 16 mappers for each case.

Looking at these tables we can see that, in general, the runtimes obtained by the Chi-FRBCS approaches are slightly lower than the ones obtained by the Chi-FRBCS-CS methods. This behavior is expected as the Chi-FRBCS-CS methods need to perform additional operations with respect to Chi-FRBCS as they include the misclassification costs in their inner way of running. Moreover, the results obtained show that the sequential versions are notably slower than the big data alternatives, even when they are compared with the performance of the big data versions on 2 mappers, as the speed gain is not linearly related to the number of mappers considered. Furthermore, this trend can also be seen among the different number of mappers considered, as the decrement in the running time is reduced meaningfully when the number of mappers is increased. This reduction in the processing time is again not lineal, as this decrement in time is more tangible at the beginning with a lower number of mappers than with a larger number of mappers.

When analyzing the behavior of the groups of cases of study derived from the original datasets we can find different groups of behavior for the cases under consideration. A first group corresponds to the bigger cases of study, the ones derived from the KDD Cup 1999 dataset and the RLCP dataset. In this case, we can see that the general trend perfectly applies to this data: the sequential versions provide runtimes that greatly exceed the results obtained by the MapReduce designs. Furthermore, the usage of higher number of mappers is able to improve the execution times, however, that

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 11
Average number of minority and majority class instances by mapper for the Chi-FRBCS-BigData and Chi-FRBCS-BigDataCS versions.

Datasets	Chi-FRBCS-BigData and Chi-FRBCS-BigDataCS							
	8 mappers		16 mappers		32 mappers		64 mappers	
	#min class	#maj class	#min class	#maj class	#min class	#maj class	#min class	#maj class
kddcup_DOS_versus_normal	97 254.60	388 360.50	48 627.30	194 180.25	24 313.65	97 090.13	12 156.83	48 545.06
kddcup_DOS_versus_PRB	4120.40	388 326.80	2060.20	194 163.40	1030.10	97 081.70	515.05	48 540.85
kddcup_DOS_versus_R2L	112.20	388 337.40	56.10	194 168.70	28.05	97 084.35	14.03	48 542.18
kddcup_DOS_versus_U2R	4.80	388 337.40	2.40	194 168.70	1.20	97 084.35	0.60	48 542.18
kddcup_normal_versus_PRB	4076.20	97 312.10	2038.10	48 656.05	1019.05	24 328.03	509.53	12 164.02
kddcup_normal_versus_R2L	117.20	97 273.50	58.60	48 636.75	29.30	24 318.38	14.65	12 159.19
kddcup_normal_versus_U2R	4.10	97 279.20	2.05	48 639.60	1.03	24 319.80	0.51	12 159.90
poker_0_vs_2	4933.30	51 319.70	2466.65	25 659.85	1233.32	12 829.93	616.66	6414.96
poker_0_vs_3	2151.60	51 382.00	1075.80	25 691.00	537.90	12 845.50	268.95	6422.75
poker_0_vs_4	398.30	51 369.70	199.15	25 684.85	99.57	12 842.43	49.79	6421.21
poker_0_vs_5	210.40	51 364.80	105.20	25 682.40	52.60	12 841.20	26.30	6420.60
poker_0_vs_6	152.00	51 364.20	76.00	25 682.10	38.00	12 841.05	19.00	6420.52
poker_0_vs_7	22.60	51 371.20	11.30	25 685.60	5.65	12 842.80	2.83	6421.40
poker_0_vs_8	1.90	51 370.00	0.95	25 685.00	0.48	12 842.50	0.24	6421.25
poker_0_vs_9	0.80	51 370.20	0.40	25 685.10	0.20	12 842.55	0.10	6421.27
poker_1_vs_2	4863.20	43 329.30	2431.60	21 664.65	1215.80	10 832.32	607.90	5416.16
poker_1_vs_3	2162.80	43 310.30	1081.40	21 655.15	540.70	10 827.57	270.35	5413.79
poker_1_vs_4	394.10	43 313.40	197.05	21 656.70	98.52	10 828.35	49.26	5414.17
poker_1_vs_5	197.90	43 316.80	98.95	21 658.40	49.47	10 829.20	24.74	5414.60
poker_1_vs_6	142.40	43 313.30	71.20	21 656.65	35.60	10 828.32	17.80	5414.16
poker_1_vs_7	24.50	43 308.80	12.25	21 654.40	6.12	10 827.20	3.06	5413.60
poker_1_vs_8	2.00	43 309.40	1.00	21 654.70	0.50	10 827.35	0.25	5413.67
poker_1_vs_9	1.20	43 309.30	0.60	21 654.65	0.30	10 827.32	0.15	5413.66
RLCP	2097.60	572 815.60	1048.80	286 407.80	524.40	143 203.90	262.20	71 601.95

Table 12
Runtime elapsed in seconds and in the hh:mm:ss.SSS format for the Chi-FRBCS versions.

Datasets	Chi-FRBCS		Chi-FRBCS-BigData									
			2 mappers		4 mappers		6 mappers		8 mappers		16 mappers	
	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS
kddcup_10_DOS_versus_normal	33 670.214	9:21:10.214	14 472.846	4:01:12.846	4140.643	1:09:00.643	1778.238	0:29:38.238	1052.501	0:17:32.501	373.428	**0:06:13.428**
kddcup_10_DOS_versus_PRB	22 510.587	6:15:10.587	9205.046	2:33:25.046	2547.166	0:42:27.166	1165.548	0:19:25.548	704.896	0:11:44.896	249.251	**0:04:09.251**
kddcup_10_DOS_versus_R2L	25 448.700	7:04:08.700	9234.977	2:33:54.977	2492.280	0:41:32.280	1197.892	0:19:57.892	707.083	0:11:47.083	228.386	**0:03:48.386**
kddcup_10_normal_versus_PRB	1756.513	0:29:16.513	1068.581	0:17:48.581	234.787	0:03:54.787	136.951	0:02:16.951	99.986	0:01:39.986	67.127	**0:01:07.127**
kddcup_10_normal_versus_R2L	1599.831	0:26:39.831	875.615	0:14:35.615	215.648	0:03:35.648	128.611	0:02:08.611	94.273	0:01:34.273	62.629	**0:01:02.629**
Average (kddcup)	16 997.169	4:43:17.169	6971.413	1:56:11.413	1926.105	0:32:06.105	881.448	0:14:41.448	531.748	0:08:51.748	196.164	**0:03:16.164**
poker_10_0_vs_2	1022.735	0:17:02.735	586.696	0:09:46.696	563.302	0:09:23.302	519.465	**0:08:39.465**	562.220	0:09:22.220	624.649	0:10:24.649
poker_10_0_vs_3	832.182	0:13:52.182	575.673	0:09:35.673	515.892	**0:08:35.892**	543.217	0:09:03.217	539.140	0:08:59.140	579.760	0:09:39.760
poker_10_0_vs_4	798.369	0:13:18.369	605.735	0:10:05.735	504.460	**0:08:24.460**	552.799	0:09:12.799	610.555	0:10:10.555	567.566	0:09:27.566
poker_10_0_vs_5	1209.566	0:20:09.566	596.255	0:09:56.255	522.352	0:08:42.352	512.363	0:08:32.363	507.853	0:08:27.853	504.863	**0:08:24.863**
poker_10_0_vs_6	1051.263	0:17:31.263	682.470	0:11:22.470	614.844	0:10:14.844	536.008	0:08:56.008	581.397	0:09:41.397	459.080	**0:07:39.080**
poker_10_0_vs_7	963.291	0:16:03.291	520.087	0:08:40.087	460.601	**0:07:40.601**	481.295	0:08:01.295	479.882	0:07:59.882	473.661	0:07:53.661
poker_10_1_vs_2	796.636	0:13:16.636	439.989	0:07:19.989	390.712	**0:06:30.712**	398.359	0:06:38.359	406.932	0:06:46.932	399.267	0:06:39.267
poker_10_1_vs_3	734.307	0:12:14.307	410.816	0:06:50.816	383.222	**0:06:23.222**	416.784	0:06:56.784	409.623	0:06:49.623	424.023	0:07:04.023
poker_10_1_vs_4	645.596	0:10:45.596	442.978	0:07:22.978	421.791	0:07:01.791	411.050	0:06:51.050	401.203	0:06:41.203	377.247	**0:06:17.247**
poker_10_1_vs_5	547.979	0:09:07.979	395.322	0:06:35.322	366.879	0:06:06.879	358.951	**0:05:58.951**	377.911	0:06:17.911	379.750	0:06:19.750
poker_10_1_vs_6	697.428	0:11:37.428	393.996	0:06:33.996	366.409	0:06:06.409	370.342	0:06:10.342	366.044	0:06:06.044	360.484	**0:06:00.484**
poker_10_1_vs_7	690.171	0:11:30.171	381.735	0:06:21.735	363.016	0:06:03.016	353.859	0:05:53.859	347.198	**0:05:47.198**	353.263	0:05:53.263
Average (poker)	832.460	0:13:52.460	502.646	0:08:22.646	456.123	0:07:36.123	454.541	**0:07:34.541**	465.830	0:07:45.830	458.634	0:07:38.634
RLCP_10	8683.187	2:24:43.187	4420.900	1:13:40.900	1174.823	0:19:34.823	562.273	0:09:22.273	369.080	0:06:09.080	179.112	**0:02:59.112**
Total average	5758.809	1:35:58.809	2517.207	0:41:57.207	904.379	0:15:04.379	579.111	0:09:39.111	478.765	0:07:58.765	370.197	**0:06:10.197**

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 13
Runtime elapsed in seconds and in the hh:mm:ss.SSS format for the Chi-FRBCS cost-sensitive versions.

Datasets	Chi-FRBCS-CS		Chi-FRBCS-BigDataCS									
			2 mappers		4 mappers		6 mappers		8 mappers		16 mappers	
	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS
kddcup_10_DOS_versus_normal	32 892.936	9:08:12.936	15 414.766	4:16:54.766	4162.916	1:09:22.916	1843.283	0:30:43.283	1076.427	0:17:56.427	415.060	**0:06:55.060**
kddcup_10_DOS_versus_PRB	29 981.173	8:19:41.173	9798.818	2:43:18.818	2585.928	0:43:05.928	1188.691	0:19:48.691	758.516	0:12:38.516	250.027	**0:04:10.027**
kddcup_10_DOS_versus_R2L	25 498.727	7:04:58.727	9649.161	2:40:49.161	2543.190	0:42:23.190	1206.790	0:20:06.790	673.231	0:11:13.231	266.537	**0:04:26.537**
kddcup_10_normal_versus_PRB	1730.916	0:28:50.916	1066.950	0:17:46.950	236.391	0:03:56.391	139.344	0:02:19.344	101.652	0:01:41.652	68.790	**0:01:08.790**
kddcup_10_normal_versus_R2L	1578.991	0:26:18.991	1072.229	0:17:52.229	212.853	0:03:32.853	134.370	0:02:14.370	94.847	0:01:34.847	65.116	**0:01:05.116**
Average (kddcup)	18 336.549	5:05:36.549	7400.385	2:03:20.385	1948.256	0:32:28.256	902.496	0:15:02.496	540.934	0:09:00.934	213.106	**0:03:33.106**
poker_10_0_vs_2	1804.004	0:30:04.004	777.028	0:12:57.028	552.463	**0:09:12.463**	557.443	0:09:17.443	579.395	0:09:39.395	572.595	0:09:32.595
poker_10_0_vs_3	1315.612	0:21:55.612	641.281	0:10:41.281	609.231	0:10:09.231	516.789	0:08:36.789	507.824	**0:08:27.824**	558.286	0:09:18.286
poker_10_0_vs_4	1630.399	0:27:10.399	764.142	0:12:44.142	607.250	0:10:07.250	582.549	0:09:42.549	581.717	0:09:41.717	572.004	**0:09:32.004**
poker_10_0_vs_5	1234.801	0:20:34.801	636.422	0:10:36.422	568.459	0:09:28.459	551.079	0:09:11.079	546.558	0:09:06.558	534.967	**0:08:54.967**
poker_10_0_vs_6	1456.625	0:24:16.625	828.762	0:13:48.762	652.770	0:10:52.770	659.502	0:10:59.502	620.019	0:10:20.019	470.053	**0:07:50.053**
poker_10_0_vs_7	1778.488	0:29:38.488	638.797	0:10:38.797	510.469	0:08:30.469	479.955	**0:07:59.955**	489.636	0:08:09.636	485.763	0:08:05.763
poker_10_1_vs_2	1137.676	0:18:57.676	496.250	0:08:16.250	425.849	0:07:05.849	396.363	**0:06:36.363**	396.681	0:06:36.681	397.112	0:06:37.112
poker_10_1_vs_3	1116.075	0:18:36.075	464.625	0:07:44.625	371.895	0:06:11.895	350.017	0:05:50.017	349.021	**0:05:49.021**	443.020	0:07:23.020
poker_10_1_vs_4	1220.649	0:20:20.649	498.319	0:08:18.319	403.404	0:06:43.404	385.980	0:06:25.980	398.769	0:06:38.769	368.581	**0:06:08.581**
poker_10_1_vs_5	1318.547	0:21:58.547	446.729	0:07:26.729	367.710	**0:06:07.710**	367.759	0:06:07.759	369.085	0:06:09.085	380.618	0:06:20.618
poker_10_1_vs_6	1453.041	0:24:13.041	478.021	0:07:58.021	394.152	0:06:34.152	377.446	0:06:17.446	351.418	**0:05:51.418**	362.195	0:06:02.195
poker_10_1_vs_7	1124.129	0:18:44.129	499.303	0:08:19.303	385.227	0:06:25.227	373.933	0:06:13.933	367.799	0:06:07.799	362.491	**0:06:02.491**
Average (poker)	1382.504	0:23:02.504	597.473	0:09:57.473	487.407	0:08:07.407	466.568	0:07:46.568	463.160	0:07:43.160	458.974	**0:07:38.974**
RLCP_10	8159.285	2:15:59.285	4165.285	1:09:25.285	1154.706	0:19:14.706	505.405	0:08:25.405	295.675	0:04:55.675	188.454	**0:03:08.454**
Total average	6468.449	1:47:48.449	2685.383	0:44:45.383	930.270	0:15:30.270	589.817	0:09:49.817	475.459	0:07:55.459	375.648	**0:06:15.648**

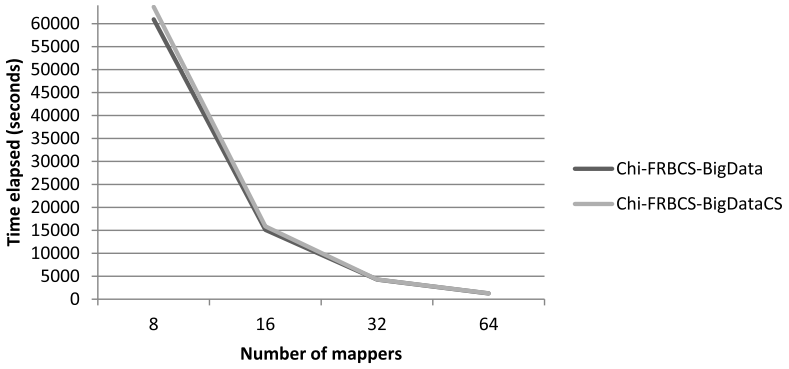


Fig. 4. Execution times for the *kddcup_full_DOS_versus_U2R* dataset.

advance is better observed when the number of mappers is smaller than in the larger cases, that is, when the data available per mapper is considerable.

The second group is related to the Poker Hand cases of study, where the processing time gain is not as clear as in the previous cases. Without a doubt, we can state there are huge differences between the sequential versions and the Hadoop implementations. When the big data versions are compared, the runtime improvement can only be detected for the smaller values of the number of mappers. The Chi-FRBCS-BigData and Chi-FRBCS-BigDataCS algorithms seem to no further improve their behavior starting from 16 mappers.

Table 14 shows the average runtime spent in seconds and in the hh:mm:ss.SSS format for the full cases of study by the Chi-FRBCS-BigData and Chi-FRBCS-BigDataCS algorithms. This table is organized in two big parts: the first part is related to the results obtained by the Chi-FRBCS-BigData algorithm while the second part is related to the Chi-FRBCS-BigDataCS method. Similarly to the preceding tables, these algorithms present their information in four columns related to the number of mappers considered: 8, 16, 32 and 64 respectively. The sequential versions are not included in this table as they are not able to provide a result, as it was shown in the scalability studies (Sections 3.3 and 4.2).

In this table, we can observe that the Chi-FRBCS-BigData approach shows a trend that slightly benefits its runtime, however, it does not always surpass the runtime achieved by the Chi-FRBCS-BigDataCS algorithm for any number of mappers. These results can be understood in the following manner: the Chi-FRBCS-BigData approach is a less complex approach than the Chi-FRBCS-BigDataCS method and therefore, the second algorithm is bound to spend more processing time due to its additional operations. The usage of cost-sensitive learning is thus a good alternative as this time addition is insignificant compared to the performance improvement gained in imbalanced datasets. In Fig. 4, we can see the difference between the performance of the big data alternatives for the *kddcup_full_DOS_versus_U2R* dataset, where the Chi-FRBCS-BigDataCS version consumes a bit more of time. However, the Chi-FRBCS-BigDataCS tends to produce a lesser number of rules (scalability studies in Sections 3.3 and 4.2), and therefore the search for identical rules may also be less computationally demanding.

In general, when larger values for the number of mappers are used, better runtime results are obtained for both the Chi-FRBCS-BigData and the Chi-FRBCS-BigDataCS algorithms. However, the improvement in the processing times is not linearly related to the number of mappers, as smaller number of mappers show a greater performance gain than larger values of mappers.

If we analyze the behavior of the groups of cases of study derived from the original datasets we can also observe the same groups of behavior as in the reduced cases of study previously considered. Again, a first group corresponds to the bigger cases of study, the ones derived from the KDD Cup 1999 dataset and the RLCP dataset. This group displays the general trend extracted from all the data: the usage of higher number of mappers can get faster execution times, however, the runtime improvement is better appreciated with a reduced number of mappers instead of with larger values, that means, when the data available per mapper is abundant. Fig. 4 also presents the trend in the usage of different mappers.

The second group is related to the Poker Hand cases of study, where it is not possible to discern an improvement in the processing times. For the smaller values of the number of mappers, the results obtained show equivalent results,

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Table 14
Runtime elapsed in seconds and in the hh:mm:ss.SSS format for the big data Chi-FRBCS versions.

Datasets	Chi-FRBCS-BigData								Chi-FRBCS-BigDataCS							
	8 mappers		16 mappers		32 mappers		64 mappers		8 mappers		16 mappers		32 mappers		64 mappers	
	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS	seconds	hh:mm:ss.SSS
kddcup_DOS_versus_normal	95 135.040	26:25:35.040	26 422.546	7:20:22.546	9678.697	2:41:18.697	4060.908	1:07:40.908	96 833.551	26:53:53.551	25 824.469	7:10:24.469	7692.155	2:08:12.155	3407.801	**0:56:47.801**
kddcup_DOS_versus_PRB	62 034.217	17:13:54.217	17 206.961	4:46:46.961	5336.043	1:28:56.043	5310.406	1:28:30.406	64 827.368	18:00:27.368	18 003.649	5:00:03.649	6094.751	1:41:34.751	5134.697	**1:25:34.697**
kddcup_DOS_versus_R2L	60 908.738	16:55:08.738	15 615.652	4:20:15.652	6315.864	1:45:15.864	1789.012	**0:29:49.012**	62 059.365	17:14:19.365	16 897.451	4:41:37.451	6122.615	1:42:02.615	2047.410	0:34:07.410
kddcup_DOS_versus_U2R	60 942.589	16:55:42.589	15 114.415	4:11:54.415	4288.956	1:11:28.956	1266.369	**0:21:06.369**	63 665.339	17:41:05.339	15 870.638	4:24:30.638	4302.037	1:11:42.037	1281.801	0:21:21.801
kddcup_normal_versus_PRB	6059.310	1:40:59.310	1765.673	0:29:25.673	548.857	0:09:08.857	262.545	**0:04:22.545**	6155.110	1:42:35.110	1523.940	0:25:23.940	753.089	0:12:33.089	301.214	0:05:01.214
kddcup_normal_versus_R2L	4362.339	1:12:42.339	1807.435	0:30:07.435	451.027	0:07:31.027	279.080	**0:04:39.080**	4502.856	1:15:02.856	1274.320	0:21:14.320	503.814	0:08:23.814	329.662	0:05:29.662
kddcup_normal_versus_U2R	4279.778	1:11:19.778	1899.410	0:31:39.410	597.729	0:09:57.729	350.514	0:05:50.514	5064.459	1:24:24.459	1290.801	0:21:30.801	730.109	0:12:10.109	327.904	**0:05:27.904**
Average (kddcup)	41 960.287	11:39:20.287	11 404.584	3:10:04.584	3888.167	1:04:48.167	1902.691	0:31:42.691	43 301.150	12:01:41.150	11 526.467	3:12:06.467	3742.653	1:02:22.653	1832.927	**0:30:32.927**
poker_0_vs_2	12 320.901	3:25:20.901	12 325.506	3:25:25.506	12 839.608	3:33:59.608	13 612.564	3:46:52.564	12 506.996	3:28:26.996	12 083.205	**3:21:23.205**	12 851.936	3:34:11.936	13 292.345	3:41:32.345
poker_0_vs_3	11 401.855	3:10:01.855	11 659.858	3:14:19.858	12 448.827	3:27:28.827	13 212.002	3:40:12.002	11 484.098	3:11:24.098	11 393.884	**3:09:53.884**	12 059.245	3:20:59.245	12 349.548	3:25:49.548
poker_0_vs_4	11 093.366	**3:04:53.366**	11 244.520	3:07:24.520	12 155.162	3:22:35.162	12 350.617	3:25:50.617	11 161.645	3:06:01.645	11 380.513	3:09:40.513	12 096.645	3:21:36.645	12 087.170	3:21:27.170
poker_0_vs_5	10 947.363	3:02:27.363	10 870.675	3:01:10.675	11 926.168	3:18:46.168	12 341.788	3:25:41.788	11 370.870	3:09:30.870	10 810.724	**3:00:10.724**	11 944.131	3:19:04.131	12 262.003	3:24:22.003
poker_0_vs_6	10 977.253	**3:02:57.253**	11 041.819	3:04:01.819	11 674.877	3:14:34.877	12 194.412	3:23:14.412	11 344.606	3:09:04.606	11 260.872	3:07:40.872	11 915.543	3:18:35.543	11 807.885	3:16:47.885
poker_0_vs_7	10 971.631	**3:02:51.631**	11 158.933	3:05:58.933	11 778.574	3:16:18.574	12 228.561	3:23:48.561	11 851.595	3:17:31.595	11 624.443	3:13:44.443	11 963.442	3:19:23.442	11 887.682	3:18:07.682
poker_0_vs_8	11 040.804	**3:04:00.804**	11 088.482	3:04:48.482	11 615.557	3:13:35.557	12 280.418	3:24:40.418	11 790.836	3:16:30.836	11 227.721	3:07:07.721	11 679.059	3:14:39.059	11 809.133	3:16:49.133
poker_0_vs_9	11 059.629	**3:04:19.629**	11 130.037	3:05:30.037	12 039.400	3:20:39.400	11 956.152	3:19:16.152	11 386.511	3:09:46.511	11 681.637	3:14:41.637	11 977.673	3:19:37.673	12 152.204	3:22:32.204
poker_1_vs_2	10 502.985	2:55:02.985	10 592.520	2:56:32.520	10 823.188	3:00:23.188	11 550.568	3:12:30.568	10 256.908	**2:50:56.908**	10 395.012	2:53:15.012	10 769.729	2:59:29.729	11 198.647	3:06:38.647
poker_1_vs_3	9734.080	2:42:14.080	10 232.695	2:50:32.695	10 770.971	2:59:30.971	10 643.134	2:57:23.134	9661.590	**2:41:01.590**	9769.442	2:42:49.442	10 434.828	2:53:54.828	10 584.726	2:56:24.726
poker_1_vs_4	9362.164	**2:36:02.164**	9599.178	2:39:59.178	9981.443	2:46:21.443	10 553.633	2:55:53.633	9443.253	2:37:23.253	9752.424	2:42:32.424	9765.667	2:42:45.667	9806.559	2:43:26.559
poker_1_vs_5	9298.083	**2:34:58.083**	9637.974	2:40:37.974	10 428.014	2:53:48.014	10 399.248	2:53:19.248	9412.589	2:36:52.589	9506.839	2:38:26.839	9942.829	2:45:42.829	10 262.902	2:51:02.902
poker_1_vs_6	9009.779	**2:30:09.779**	9591.369	2:39:51.369	10 112.862	2:48:32.862	10 407.752	2:53:27.752	9739.623	2:42:19.623	9854.607	2:44:14.607	9963.349	2:46:03.349	10 095.291	2:48:15.291
poker_1_vs_7	9285.360	2:34:45.360	9250.462	**2:34:10.462**	9962.175	2:46:02.175	10 333.898	2:52:13.898	9580.927	2:39:40.927	9670.806	2:41:10.806	10 300.841	2:51:40.841	10 276.786	2:51:16.786
poker_1_vs_8	9545.055	2:39:05.055	9380.564	**2:36:20.564**	9872.084	2:44:32.084	10 226.082	2:50:26.082	9830.342	2:43:50.342	9422.569	2:37:02.569	9912.194	2:45:12.194	10 300.646	2:51:40.646
poker_1_vs_9	9179.436	**2:32:59.436**	9438.347	2:37:18.347	9893.532	2:44:53.532	10 335.326	2:52:15.326	9776.855	2:42:56.855	9844.250	2:44:04.250	10 195.108	2:49:55.108	10 476.054	2:54:36.054
Average (poker)	10 358.109	**2:52:38.109**	10 515.184	2:55:15.184	11 145.153	3:05:45.153	11 539.135	3:12:19.135	10 662.453	2:57:42.453	10 604.934	2:56:44.934	11 110.764	3:05:10.764	11 290.599	3:08:10.599
RLCP	26 551.162	7:22:31.162	7089.999	1:58:09.999	1922.670	0:32:02.670	606.831	**0:10:06.831**	27 547.418	7:39:07.418	7270.635	2:01:10.635	1830.273	0:30:30.273	721.305	0:12:01.305
Final average	20 250.122	5:37:30.122	10 631.876	2:57:11.876	8644.262	2:24:04.262	8272.992	2:17:52.992	20 885.613	5:48:05.613	10 734.785	2:58:54.785	8575.044	2:22:55.044	8091.724	**2:14:51.724**

however, when larger values of mappers are considered, the runtime does not improve and it can even become worse. This situation arises due to the smaller size of the Poker Hand cases of study.

Finally, it is necessary to recall that even when a larger number of mappers tend to provide better response times it may not be wise to try to expand that number as much as possible. As we observed in Section 6.3.1, a large number of mappers may cause a dramatically drop in the performance, an unwanted case when trying to extract information from data. Therefore, it is needed to analyze the case under consideration to select an appropriate number of mappers for the experiment. This number of mappers needs to provide a reasonable number of samples for each class to avoid the small sample size problem and also enough data so that the experiments obtain lesser response times.

To sum up, our experimental study shows that cost-sensitive learning allows us to obtain better classification results for the Chi-FRBCS algorithm. We have also observed that, in the big data versions, increasing the number of mappers decreases the accuracy of the model, not only because the full information is not available but also because of the induction of data intrinsic problems that difficult the classification with imbalanced datasets, such as the small sample size problem. Finally, big data versions allow us to deal with huge amounts of data and obtain better response times which are generally significantly decremented when the number of mappers of the original dataset is increased.

7. Concluding remarks

In this paper, we have introduced a linguistic cost-sensitive fuzzy rule-based classification method for imbalanced big data called Chi-FRBCS-BigDataCS. Our aim was to obtain a model that is able to handle imbalanced big data obtaining a good precision without incrementing the processing times. To do so, we use one of the most popular approaches nowadays to deal with big data: the MapReduce framework, distributing the algorithm computing along different processing units using the map and reduce operations that have been adapted to the calculations of the fuzzy rule based classification system. We have also used cost-sensitive learning operations which have also modified the algorithm to consider the misclassification costs, proposing a new approach, PCF-CS, to compute the rule weight that consider these costs in its operations.

The experiments conducted in this work demonstrate that the MapReduce framework is able of dealing with big data for fuzzy rule based classification systems. The use of a simple but effective fuzzy rule based classification system such as the Chi et al.'s method as base of the approach has enabled the development of a proposal that can profit from this simplicity to create an efficient approach. The proposal, Chi-FRBCS-BigDataCS, can obtain classification results when its sequential counterpart was not able to provide results. Furthermore, the runtime needed by the proposal is admissible according to the results presented. The inclusion of cost-sensitive learning in its way of working, using the new rule weight procedure PCF-CS, has demonstrated to be a powerful collaborator when dealing with imbalanced datasets providing effective classification results without largely increasing the processing times.

The performance of our model, Chi-FRBCS-BigDataCS, has been tested in an experimental study including twenty-four imbalanced big data cases of study. These results corroborate the goodness of the integration of the approaches that are used to solve the imbalanced problem and big data separately, namely the usage of the MapReduce framework and cost-sensitive learning. Furthermore, the synergy between both strategies alleviates some data intrinsic problems, like the small sample size problem, that are induced because of the way the learning is done.

Acknowledgements

This work was partially supported by the Spanish Ministry of Science and Technology under project TIN2011-28488 and the Andalusian Research Plans P11-TIC-7765 and P10-TIC-6858. V. López holds a FPU scholarship from Spanish Ministry of Education.

References

[1] IBM, What is big data? Bringing big data to the enterprise, [Online; accessed December 2013], http://www-01.ibm.com/software/data/bigdata/, 2012.

[2] P. Zikopoulos, C. Eaton, D. DeRoos, T. Deutsch, G. Lapis, Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data, McGraw-Hill, 2011.

[3] S. Madden, From databases to big data, IEEE Internet Comput. 16 (3) (2012) 4–6.

[4] A. Sathi, Big Data Analytics: Disruptive Technologies for Changing the Game, MC Press, 2012.

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

[5] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[6] Y. Sun, A.K.C. Wong, M.S. Kamel, Classification of imbalanced data: A review, Int. J. Pattern Recognit. Artif. Intell. 23 (4) (2009) 687–719.
[7] V. López, A. Fernández, S. García, V. Palade, F. Herrera, An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics, Inf. Sci. 250 (2013) 113–141.
[8] H. Ishibuchi, T. Nakashima, M. Nii, Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining, Springer-Verlag, 2004.
[9] Y. Jin, Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement, IEEE Trans. Fuzzy Syst. 8 (2) (2000) 212–221.
[10] T.-P. Hong, Y.-C. Lee, M.-T. Wu, An effective parallel approach for genetic-fuzzy data mining, Expert Syst. Appl. 41 (2) (2014) 655–662.
[11] M. Rodríguez, D. Escalante, A. Peregrín, Efficient distributed genetic algorithm for rule extraction, Appl. Soft Comput. 11 (1) (2011) 733–743.
[12] Y. Nojima, H. Ishibuchi, I. Kuwajima, Parallel distributed genetic fuzzy rule selection, Soft Comput. 13 (5) (2009) 511–519.
[13] I. Robles, R. Alcalá, J. Benítez, F. Herrera, Evolutionary parallel and gradually distributed lateral tuning of fuzzy rule-based systems, Evol. Intel. 2 (1–2) (2009) 5–19.
[14] H. Ishibuchi, S. Mihara, Y. Nojima, Parallel distributed hybrid fuzzy GBML models with rule set migration and training data rotation, IEEE Trans. Fuzzy Syst. 21 (2) (2013) 355–368.
[15] V. López, A. Fernández, J.G. Moreno-Torres, F. Herrera, Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics, Expert Syst. Appl. 39 (7) (2012) 6585–6608.
[16] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer SMOTE, Synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[17] G.E.A.P.A. Batista, R.C. Prati, M.C. Monard, A study of the behaviour of several methods for balancing machine learning training data, SIGKDD Explor. 6 (1) (2004) 20–29.
[18] C. Elkan, The foundations of cost–sensitive learning, in: Proceedings of the 17th IEEE International Joint Conference on Artificial Intelligence (IJCAI'01), 2001, pp. 973–978.
[19] B. Zadrozny, C. Elkan, Learning and making decisions when costs and probabilities are both unknown, in: Proceedings of the 7th International Conference on Knowledge Discovery and Data Mining (KDD'01), 2001, pp. 204–213.
[20] J. Dean, S. Ghemawat, MapReduce: Simplified data processing on large clusters, Commun. ACM 51 (1) (2008) 107–113.
[21] M. Wasikowski, X.-W. Chen, Combating the small sample class imbalance problem using feature selection, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1388–1400.
[22] J.G. Moreno-Torres, T. Raeder, R. Aláiz-Rodríguez, N.V. Chawla, F. Herrera, A unifying view on dataset shift in classification, Pattern Recognit. 45 (1) (2012) 521–530.
[23] Z. Chi, H. Yan, T. Pham, Fuzzy Algorithms with Applications to Image Processing and Pattern Recognition, World Scientific, 1996.
[24] T. Nakashima, G. Schaefer, Y. Yokota, H. Ishibuchi, Weighted fuzzy classifier and its application to image processing tasks, Fuzzy Sets Syst. 158 (2007) 284–294.
[25] V. López, A. Fernández, M.J. del Jesus, F. Herrera, A hierarchical genetic fuzzy system based on genetic programming for addressing classification with highly imbalanced and borderline data-sets, Knowl.-Based Syst. 38 (2013) 85–104.
[26] A. Fernández, S. García, M.J. del Jesus, F. Herrera, A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets, Fuzzy Sets Syst. 159 (18) (2008) 2378–2398.
[27] A. Fernández, M.J. del Jesus, F. Herrera, Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets, Int. J. Approx. Reason. 50 (3) (2009) 561–577.
[28] K. Napierala, J. Stefanowski, S. Wilk, Learning from imbalanced data in presence of noisy and borderline examples, in: Proceedings of the 7th International Conference on Rough Sets and Current Trends in Computing, RSCTC'10, in: Lecture Notes on Artificial Intelligence, vol. 6086, 2010, pp. 158–167.
[29] J.A. Sáez, J. Luengo, F. Herrera, A first study on the noise impact in classes for fuzzy rule based classification systems, in: Proceedings of the 2010 IEEE International Conference on Intelligent Systems and Knowledge Engineering, ISKE'10, IEEE Press, 2010, pp. 153–158.
[30] G.M. Weiss, The impact of small disjuncts on classifier learning, in: R. Stahlbock, S.F. Crone, S. Lessmann (Eds.), Data Mining, in: Annals of Information Systems, vol. 8, Springer, 2010, pp. 193–226.
[31] J. Huang, C.X. Ling, Using AUC and accuracy in evaluating learning algorithms, IEEE Trans. Knowl. Data Eng. 17 (3) (2005) 299–310.
[32] T. White, Hadoop, The Definitive Guide, O'Reilly Media, Inc., 2012.
[33] D. Laney, 3D data management: Controlling data volume, velocity, and variety, META Group, 2001, Tech. rep., [Online; accessed December 2013], http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf.
[34] M. Beyer, Gartner says solving big data challenge involves more than just managing volumes of data, [Online; accessed December 2013], 2011, http://www.gartner.com/newsroom/id/1731916.
[35] M. Beyer, D. Laney, The importance of big data: A definition, ID: G00235055, Retrieved from Gartner database [Online; accessed December 2013], 2012, http://www.gartner.com/id=2057415.
[36] J. Dean, S. Ghemawat, MapReduce: Simplified data processing on large clusters, in: Proceedings of the 6th Symposium on Operating System Design and Implementation, OSDI 2004, 2004, pp. 137–150.
[37] J. Dean, S. Ghemawat, MapReduce: A flexible data processing tool, Commun. ACM 53 (1) (2010) 72–77.
[38] C. Lam, Hadoop in Action, Manning Publications Co., 2010.
[39] S. Owen, R. Anil, T. Dunning, E. Friedman, Mahout in Action, Manning Publications Co., 2011.
[40] J. Lin, MapReduce is good enough? If all you have is a hammer, throw away everything that's not a nail!, Big Data 1 (1) (2013) 28–37.
[41] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. Franklin, S. Shenker, I. Stoica, Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing, in: Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2012, 2012.

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

[42] Apache Drill Project, Apache Drill, 2013, [Online; December 2013, accessed], http://incubator.apache.org/drill/.
[43] J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.-H. Bae, J. Qiu, G. Fox, Twister: a runtime for iterative MapReduce, in: Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing (HPDC 2010), 2010, pp. 810–818.
[44] S. Das, Y. Sismanis, K.S. Beyer, R. Gemulla, P.J. Haas, J. McPherson, Ricardo: integrating R and Hadoop, in: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2010), 2010, pp. 987–998.
[45] M. Bostock, O.V., J. Heer, D3 data-driven documents, IEEE Trans. Vis. Comput. Graph. 17 (12) (2011) 2301–2309.
[46] HCatalog, Hcatalog, [Online; accessed December 2013, accessed] http://hive.apache.org/hcatalog/ (2013).
[47] J. Leibiusky, G. Eisbruch, D. Simonassi, Getting Started with Storm, O'Reilly Media, Inc., 2012.
[48] Cloudera, Cloudera Impala, [Online; accessed December 2013] (2013). http://www.cloudera.com/content/cloudera/en/products/cdh/impala.html.
[49] Q. Yang, X. Wu, 10 challenging problems in data mining research, Int. J. Inf. Technol. Decis. Mak. 5 (4) (2006) 597–604.
[50] T. Khoshgoftaar, K. Gao, A. Napolitano, R. Wald, A comparative study of iterative and non-iterative feature selection techniques for software defect prediction, Inf. Syst. Front., in press, http://dx.doi.org/10.1007/s10796-013-9430-0.
[51] S. Wang, X. Yao, Using class imbalance learning for software defect prediction, IEEE Trans. Reliab. 62 (2) (2013) 434–443.
[52] L. Zhou, Performance of corporate bankruptcy prediction models on imbalanced dataset: The effect of sampling methods, Knowl.-Based Syst. 41 (2013) 16–25.
[53] A. Gudys, M. Szczesniak, M. Sikora, I. Makalowska, HuntMi: An efficient and taxon-specific approach in pre-miRNA identification, BMC Bioinform. 14 (2013) 1–10, Article number 83.
[54] Q. Wei, R. Dunbrack Jr., The role of balanced training and testing data sets for binary classifiers in bioinformatics, PLoS ONE 8 (7) (2013) 1–12, Article number e67863.
[55] H. Yu, J. Ni, J. Zhao, ACOSampling: An ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data, Neurocomputing 101 (2013) 309–318.
[56] Y.-H. Lee, P. Hu, T.-H. Cheng, T.-C. Huang, W.-Y. Chuang, A preclustering-based ensemble learning technique for acute appendicitis diagnoses, Artif. Intell. Med. 58 (2) (2013) 115–124.
[57] J. Nahar, T. Imam, K. Tickle, Y.-P. Chen, Computational intelligence for heart disease diagnosis: A medical knowledge driven approach, Expert Syst. Appl. 40 (1) (2013) 96–104.
[58] A. Orriols-Puig, E. Bernadó-Mansilla, Evolutionary rule-based systems for imbalanced datasets, Soft Comput. 13 (3) (2009) 213–225.
[59] V. García, R.A. Mollineda, J.S. Sánchez, On the k-NN performance in a challenging scenario of imbalance and overlapping, Pattern Anal. Appl. 11 (3–4) (2008) 269–280.
[60] C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, A. Folleco, An empirical study of the classification performance of learners on imbalanced and noisy software quality data, Inf. Sci. 259 (2014) 571–595.
[61] J. Stefanowski, Overlapping, rare examples and class decomposition in learning classifiers from imbalanced data, in: Smart Innovation, Systems and Technologies, vol. 13, 2013, pp. 277–306.
[62] A. Storkey, When training and test sets are different: Characterizing learning transfer, in: J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, N.D. Lawrence (Eds.), Dataset Shift in Machine Learning, MIT Press, 2009, pp. 3–28.
[63] H. Shimodaira, Improving predictive inference under covariate shift by weighting the log-likelihood function, J. Stat. Plan. Inference 90 (2) (2000) 227–244.
[64] V. López, I. Triguero, C. Carmona, S. García, F. Herrera, Addressing imbalanced classification with instance generation techniques: IPADE-ID, Neurocomputing 126 (2014) 15–28.
[65] P. Domingos, MetaCost: A general method for making classifiers cost-sensitive, in: Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD'99), 1999, pp. 155–164.
[66] B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting, in: Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03), 2003, pp. 435–442.
[67] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for class imbalance problem: Bagging, boosting and hybrid based approaches, IEEE Trans. Syst. Man Cybern., Part C, Appl. Rev. 42 (4) (2012) 463–484.
[68] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognit. 30 (7) (1997) 1145–1159.
[69] H. Ishibuchi, T. Nakashima, Effect of rule weights in fuzzy rule-based classification systems, IEEE Trans. Fuzzy Syst. 9 (4) (2001) 506–515.
[70] H. Ishibuchi, T. Yamamoto, Rule weight specification in fuzzy rule-based classification systems, IEEE Trans. Fuzzy Syst. 13 (2005) 428–435.
[71] O. Cordón, M.J. del Jesus, F. Herrera, A proposal on reasoning methods in fuzzy rule-based classification systems, Int. J. Approx. Reason. 20 (1) (1999) 21–45.
[72] L.X. Wang, J.M. Mendel, Generating fuzzy rules by learning from examples, IEEE Trans. Syst. Man Cybern. 22 (6) (1992) 1414–1427.
[73] K. Bache, M. Lichman, UCI machine learning repository, [Online; accessed December 2013], 2013, http://archive.ics.uci.edu/ml.
[74] M. Fazzolari, B. Giglio, R. Alcalá, F. Marcelloni, F. Herrera, A study on the application of instance selection techniques in genetic fuzzy rule-based classification systems: Accuracy-complexity trade-off, Knowl.-Based Syst. 54 (2014) 32–41.
[75] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Comput. 13 (2009) 307–318.
[76] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework, J. Mult.-Valued Log. Soft Comput. 17 (2–3) (2011) 255–287.

Please cite this article in press as: V. López et al., Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data, Fuzzy Sets and Systems (2014), http://dx.doi.org/10.1016/j.fss.2014.01.015

Bibliography

[ACW06] Au W. H., Chan K. C. C., and Wong A. K. C. (2006) A fuzzy approach to partitioning continuous attributes for classification. *IEEE Transactions on Knowledge and Data Engineering* 18(5): 715–719.

[ADA11] Agrawal D., Das S., and Abbadi A. E. (2011) Big data and cloud computing: current state and future opportunities. In *Proceedings of the 14th International Conference on Extending Database Technology (EDBT 2011)*, pp. 530–533.

[AKA91] Aha D., Kibler D., and Albert M. (1991) Instance-based learning algorithms. *Machine Learning* 6(1): 37–66.

[Alp04] Alpaydin E. (2004) *Introduction to Machine Learning*. The MIT Press.

[BF99] Brodley C. E. and Friedl M. A. (1999) Identifying mislabeled training data. *Journal of Artificial Intelligence Research* 11: 131–167.

[BP10] Batuwita R. and Palade V. (2010) FSVM-CIL: Fuzzy support vector machines for class imbalance learning. *IEEE Transactions on Fuzzy Systems* 18(3): 558–571.

[BPM04] Batista G. E. A. P. A., Prati R. C., and Monard M. C. (2004) A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations Newsletter* 6(1): 20–29.

[BRdJH10] Berlanga F., Rivera A., del Jesus M., and Herrera F. (2010) GP-COACH: Genetic programming-based learning of compact and accurate fuzzy rule-based classification systems for high-dimensional problems. *Information Sciences* 180(8): 1183 – 1200.

[CBHK02] Chawla N. V., Bowyer K. W., Hall L. O., and Kegelmeyer W. P. (2002) SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* 16: 321–357.

[CDG+08] Chang F., Dean J., Ghemawat S., Hsieh W. C., Wallach D. A., Burrows M., Chandra T., Fikes A., and Gruber R. E. (2008) Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems* 26(2).

[CdJH99] Cordón O., del Jesus M., and Herrera F. (1999) A proposal on reasoning methods in fuzzy rule-based classification systems. *International Journal of Approximate Reasoning* 20(1): 21–45.

[CH67] Cover T. M. and Hart P. E. (1967) Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13: 21–27.

[CHV00] Cordón O., Herrera F., and Villar P. (2000) Analysis and guidelines to obtain a good uniform fuzzy partition granularity for fuzzy rule-based systems using simulated annealing. *International Journal on Approximate Reasoning* 25(3): 187–215.

[CJK04] Chawla N. V., Japkowicz N., and Kolcz A. (2004) Special issue on learning from imbalanced datasets. *SIGKDD Explorations Newsletter* 6(1): 1–6.

[CV95] Cortes C. and Vapnik V. (1995) Support vector networks. *Machine Learning* 20: 273–297.

[CW03] Chen Y. and Wang J. Z. (2003) Support vector learning for fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems* 11(6): 716–728.

[CYP96] Chi Z., Yan H., and Pham T. (1996) *Fuzzy algorithms with applications to image processing and pattern recognition.* World Scientific.

[DG08] Dean J. and Ghemawat S. (2008) MapReduce: simplified data processing on large clusters. *Commununications of the ACM* 51(1): 107–113.

[DGS+90] DeWitt D. J., Ghandeharizadeh S., Schneider D. A., Bricker A., Hsiao H. I., and Rasmussen R. (1990) The gamma database machine project. *IEEE Transactions on Knowledge and Data Engineering* 2(1): 44–62.

[DHS01] Duda R. O., Hart P. E., and Stork D. G. (2001) *Pattern Classification.* Wiley-Interscience.

[DKS09] Drown D. J., Khoshgoftaar T. M., and Seliya N. (2009) Evolutionary sampling and software quality modeling of high-assurance systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part A* 39(5): 1097–1107.

[DT10] Denil M. and Trappenberg T. (2010) Overlap versus imbalance. In *Proceedings of the 23rd Canadian Conference on advances in Artificial Intelligence (CCAI'10)*, pp. 220–231.

[FdJH09] Fernández A., del Jesus M. J., and Herrera F. (2009) Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets. *International Journal of Approximate Reasoning* 50(3): 561–577.

[FdJH10] Fernández A., del Jesus M. J., and Herrera F. (2010) On the 2–tuples based genetic tuning performance for fuzzy rule based classification systems in imbalanced data–sets. *Information Sciences* 180(8): 1268–1291.

[FGdJH08] Fernández A., García S., del Jesus M. J., and Herrera F. (2008) A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data–sets. *Fuzzy Sets and Systems* 159(18): 2378–2398.

[FGL+10] Fernández A., García S., Luengo J., Bernadó-Mansilla E., and Herrera F. (2010) Genetics-based machine learning for rule induction: State of the art, taxonomy and comparative study. *IEEE Transactions on Evolutionary Computation* 14(6): 913–941.

[FPSS96] Fayyad U., Piatetsky-Shapiro G., and Smyth P. (1996) From data mining to knowledge discovery in databases. *AI Magazine* 17(3): 37–54.

[GGM12] Gupta R., Gupta H., and Mohania M. (2012) Cloud computing and big data analytics: What is new from databases perspective? In *Proceedings of the First International Conference on Big Data Analytics (BDA 2012)*, pp. 42–61.

[GLDS96] Gropp W., Lusk E. L., Doss N. E., and Skjellum A. (1996) A high-performance, portable implementation of the mpi message passing interface standard. *Parallel Computing* 22(6): 789–828.

[GMS08] García V., Mollineda R. A., and Sánchez J. S. (2008) On the k–NN performance in a challenging scenario of imbalance and overlapping. *Pattern Analysis Applications* 11(3–4): 269–280.

[Gol89] Goldberg D. E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley.

[Gur97] Gurney K. N. (1997) *An introduction to neural networks.* Morgan Kaufmann.

[HG09] He H. and Garcia E. A. (2009) Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering* 21(9): 1263–1284.

[HHLD11] Han J., Haihong E., Le G., and Du J. (2011) Survey on NoSQL database. In *Proceedings of the 2011 6th International Conference on Pervasive Computing and Applications (ICPCA 2011)*, pp. 363–366.

[INN04] Ishibuchi H., Nakashima T., and Nii M. (2004) *Classification and modeling with linguistic information granules: Advanced approaches to linguistic Data Mining.* Springer–Verlag.

[IY05] Ishibuchi H. and Yamamoto T. (2005) Rule weight specification in fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems* 13: 428–435.

[Kon05] Konar A. (2005) *Computational Intelligence: Principles, Techniques and Applications.* Springer-Verlag.

[KR14] Kuncheva L. I. and Rodríguez J. J. (2014) A weighted voting framework for classifiers ensembles. *Knowledge and Information Systems* 38(2): 259–275.

[Kra13] Kraska T. (2013) Finding the needle in the big data systems haystack. *IEEE Internet Computing* 17(1): 84–86.

[Kun00] Kuncheva L. (2000) *Fuzzy classifier design.* Springer.

[Lam11] Lam C. (2011) *Hadoop in action.* Manning.

[LJ12] Labrinidis A. and Jagadish H. V. (2012) Challenges and opportunities with big data. *Proceedings of the VLDB Endowment* 5(12): 2032–2033.

[LTY13] Lin M., Tang K., and Yao X. (2013) Dynamic sampling approach to training neural networks for multiclass imbalance classification. *IEEE Transactions on Neural Networks and Learning Systems* 24(4): 647–660.

[LWZ09] Liu X. Y., Wu J., and Zhou Z. H. (2009) Exploratory undersampling for class-imbalance learning. *IEEE Transactions on System, Man and Cybernetics, Part B* 39(2): 539–550.

[Mad12] Madden S. (2012) From databases to big data. *IEEE Internet Computing* 16(3): 4–6.

[Mam74] Mamdani E. (1974) Applications of fuzzy algorithm for control a simple dynamic plant. *Proceedings of the Institution of Electrical Engineers* 121(12): 1585–1588.

[MTH10] Moreno-Torres J. G. and Herrera F. (2010) A preliminary study on overlapping and data fracture in imbalanced domains by means of genetic programming-based feature extraction. In *Proceedings of the 10th International Conference on Intelligent Systems Design and Applications (ISDA'10)*, pp. 501–506.

[MTSH12] Moreno-Torres J., Sáez J., and Herrera F. (2012) Study on the impact of partition-induced dataset shift on k-fold cross-validation. *IEEE Transactions on Neural Networks and Learning Systems* 23(8): 1304–1312.

[NSW10] Napierala K., Stefanowski J., and Wilk S. (2010) Learning from imbalanced data in presence of noisy and borderline examples. In *Proceedings of the 7th International Conference on Rough Sets and Current Trends in Computing (RSCTC'10)*, pp. 158–167.

[OPBM09] Orriols-Puig A. and Bernadó-Mansilla E. (2009) Evolutionary rule–based systems for imbalanced datasets. *Soft Computing* 13(3): 213–225.

[OPBMG+09] Orriols-Puig A., Bernadó-Mansilla E., Goldberg D. E., Sastry K., and Lanzi P. L. (2009) Facetwise analysis of XCS for problems with class imbalances. *IEEE Transactions on Evolutionary Computation* 13: 260–283.

[Pet07] Peters J. (2007) Book review: Computational intelligence: Principles, techniques and applications by Amit Konar. *The Computer Journal* 50(6): 758.

[Pyl99] Pyle D. (1999) *Data Preparation for Data Mining.* Morgan Kaufmann.

[Qui93] Quinlan J. R. (1993) *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

[RJBF+80] Rothnie Jr. J. B., Bernstein P. A., Fox S., Goodman N., Hammer M., Landers T. A., Reeve C. L., Shipman D. W., and Wong E. (1980) Introduction to a system for distributed databases (SDD-1). *ACM Transactions on Database Systems* 5(1): 1–17.

[SAM96] Shafer J., Agrawal R., and Mehta M. (1996) Sprint: A scalable parallel classifier for data mining. In *Proceedings of the 22th International Conference on Very Large Data Bases (VLDB'96)*, pp. 544–555.

[Shi00] Shimodaira H. (2000) Improving predictive inference under Covariate Shift by Weighting the Log-likelihood Function. *Journal of Statistical Planning and Inference* 90(2): 227–244.

[SJ12] Schlieski T. and Johnson B. D. (2012) Entertainment in the age of big data. *Proceedings of the IEEE* 100(Special Centennial Issue): 1404–1408.

[SKVHF14] Seiffert C., Khoshgoftaar T. M., Van Hulse J., and Folleco A. (2014) An empirical study of the classification performance of learners on imbalanced and noisy software quality data. *Information Sciences* 259: 571–595.

[SKVHN10] Seiffert C., Khoshgoftaar T. M., Van Hulse J., and Napolitano A. (2010) RUSBoost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on System, Man and Cybernetics, Part A* 40(1): 185–197.

[SKWW07] Sun Y., Kamel M. S., Wong A. K. C., and Wang Y. (2007) Cost–sensitive boosting for classification of imbalanced data. *Pattern Recognition* 40(12): 3358–3378.

[SLH10] Sáez J., Luengo J., and Herrera F. (2010) A first study on the noise impact in classes for fuzzy rule based classification systems. In *2010 International Conference on Intelligent Systems and Knowledge Engineering (ISKE2010)*, pp. 153–158.

[SWK09] Sun Y., Wong A. K. C., and Kamel M. S. (2009) Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence* 23(4): 687–719.

[The12] The Apache Software Foundation (2012) Hadoop, an open source implementation of MapReduce and GFS.

[TSK06] Tan P. N., Steinbach M., and Kumar V. (2006) *Introduction to Data Mining*. Addison-Wesley.

[VHKN09] Van Hulse J., Khoshgoftaar T. M., and Napolitano A. (2009) An empirical comparison of repetitive undersampling techniques. In *Proceedings of the 2009 IEEE International Conference on Information Reuse Integration (IRI'09)*, pp. 29–34.

[Wei05] Weiss G. M. (2005) Mining with rare cases. In Maimon O. and Rokach L. (Eds.) *The Data Mining and Knowledge Discovery Handbook*, pp. 765–776. Springer.

[Wei10] Weiss G. M. (2010) The impact of small disjuncts on classifier learning. In Stahlbock R., Crone S. F., and Lessmann S. (Eds.) *Data Mining*, volumen 8 of *Annals of Information Systems*, pp. 193–226. Springer.

[WY13] Wang S. and Yao X. (2013) Relationships between diversity of classification ensembles and single-class performance measures. *IEEE Transactions on Knowledge and Data Engineering* 25(1): 206–219.

[WYLD10] White B., Yeh T., Lin J., and Davis L. (2010) Web-scale computer vision using MapReduce for multimedia data mining. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining (MDMKDD'10)*, pp. 9:1–9:10.

[WZWD14] Wu X., Zhu X., Wu G. Q., and Ding W. (2014) Data mining with big data. *IEEE Transactions On Knowledge And Data Engineering* 26(1): 97–107.

[YW06] Yang Q. and Wu X. (2006) 10 challenging problems in data mining research. *International Journal of Information Technology and Decision Making* 5(4): 597–604.

[Zad65] Zadeh L. A. (1965) Fuzzy sets. *Information and Control* 8: 338–353.

[ZHC13] Zong W., Huang G. B., and Chen Y. (2013) Weighted extreme learning machine for imbalance learning. *Neurocomputing* 101: 229–242.

[ZLA03] Zadrozny B., Langford J., and Abe N. (2003) Cost–sensitive learning by cost–proportionate example weighting. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03)*, pp. 435–442.