UNIVERSIDAD DE GRANADA

Departamento de Ciencias de la Computación e Inteligencia Artificial

NUEVOS MÉTODOS DE EDICIÓN DE CONJUNTOS DE ENTRENAMIENTO NO BALANCEADOS USANDO LA TEORÍA DE LOS CONJUNTOS APROXIMADOS

Tesis Doctoral

Enislay Ramentol Martínez

Editor: Editorial de la Universidad de Granada
Autor: Enislay Ramentol Martínez
D.L.: GR 2100-2014
ISBN: 978-84-9083-128-1

UNIVERSIDAD DE GRANADA

NUEVOS MÉTODOS DE EDICIÓN DE CONJUNTOS DE ENTRENAMIENTO NO BALANCEADOS USANDO LA TEORÍA DE LOS CONJUNTOS APROXIMADOS

Memoria presentada por

Enislay Ramentol Martínez

para optar por el título de Doctor en Informática

DIRECTORES

Francisco Herrera Triguero y Rafael Bello Pérez

La doctoranda Dña. Enislay Ramentol Martínez y los directores de la tesis D.Francisco Herrera Triguero y D.Rafael Bello Pérez garantizamos, al firmar esta tesis doctoral, que el trabajo ha sido realizado por la doctoranda 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, Mayo de 2014

Directores de la Tesis

Fdo. Francisco Herrera Triguero Fdo. Rafael Bello Pérez

Doctoranda

Fdo. Enislay Ramentol Martínez

Dedicatoria

A mi esposo, mis padres, mi hermano. A quien está por venir. . .

Agradecimientos

Me gustaría agradecer a todas aquellas personas que me han ayudado a alcanzar esta meta, el camino ha parecido largo, pero finalmente aquí estamos.

Quisiera agradecer en primer lugar a mi familia, en especial a mi esposo, por su amor incondicional, por estar a mi lado cuando la meta parecía alejarse mucho, por soportar tanto tiempo separados. A mis padres porque les debo todo cuanto soy, por todo su apoyo, por servirme el café en aquellas madrugadas de experimentaciones sin poder pegar ojo. A mi hermano, porque siempre me ha dado fuerzas, porque nunca desiste, porque siempre logra sacarme una sonrisa aunque esté en el peor momento, por estar ahí.

Agradezco infinitamente a mis directores Francisco Herrera y Rafael Bello, porque cada regaño me hizo crecer, por los consejos, las palabras de ánimos, las palabras fuertes que me sacaban lágrimas, por todo el tiempo que me han dedicado, por confiar en mí, por darme la oportunidad de trabajar con dos brillantes investigadores.

Me gustaría expresar mi gratitud a Chris Cornelis, por ser igualmente mi guía, por la revisión de todos mis trabajos, por las sugerencias tan oportunas, porque siempre resultó algo bueno de cada idea que me dio, por todo el tiempo que me ha dedicado durante estos años.

A Yailé Caballero, por contribuir a mi formación desde que era estudiante de ingeniería, por haber sido mi guía, por llevarme por el camino de la investigación, por enseñarme que siempre el sacrificio es recompensado, por sus sabios consejos.

Mi agradecimiento para Sarah Vluymans y Nele Verbiest porque la colaboración con ellas fue muy provechosa, por ser buenas compañeras. Al grupo de investigación del Profesor

Paco, en especial a Alberto Fernández, Julián Luengo y Joaquín Derrac por toda la ayuda que siempre me brindaron, sobre todo las relacionadas con KEEL.

Me gustaría agradecer de manera muy especial a la AUIP y al grupo COIMBRA, porque gracias a ellos pude realizar provechosas estancias de investigación en la UGR.

A mis compañeros y amigos de la Universidad de Camagüey Yaima Filiberto, Julio Madera, Sandro Martínez, Elizabet Tejeda, Marcos Leiva, Josefina Arboleas, Geysel Salgado, Jorge Recio, a todos gracias por su amistad y su ayuda. A mis amigos de toda la vida Dalgys, Biasmey y Telvys por apoyarme y mejorarme la vida.

Agradezco además a todos los profesores que impartieron cursos en Cuba cuando el doctorado comenzó: Jorge Casillas, César Hervás, José Luis Verdegay, David Pelta, Juan Huete , Juan M. Fdez. Luna, Miguel García Silvente, Jesús Aguilar, Rafael Morales y Gonzalo Ramos.

A todos los que figuran en esta lista, y a los que no están pero igualmente me han ayudado de alguna forma a lograr esta meta, a todos:

MUCHAS GRACIAS!!!!

Resumen

En esta tesis se presenta un conjunto de algoritmos para tratar datos no balanceados. Hemos utilizado la Teoría de los Conjuntos Aproximados (TCA), tanto la clásica como su enfoque difuso (TCAD), para evaluar la pertenencia de los objetos a la región positiva de su clase, y bajo este criterio decantarlos como buenos representas de ésta. Nuestras propuestas están enfocadas en 2 niveles: al nivel de los datos, donde proponemos un grupo de algoritmos híbridos de sobremuestreo y al nivel de los algoritmos, donde proponemos un algoritmo de clasificación para conjuntos no balanceados.

Los objetivos desarrollados fueron:

Proponer un nuevo algoritmo híbrido de preprocesamiento para conjuntos de alto desbalance que utilice SMOTE para generar nuevos ejemplos y evalúe la calidad de dichos ejemplos a través de la TCA. El objetivo de esta propuesta es que cada instancia sintética que sea generada con SMOTE pertenezca a la aproximación inferior de la clase, de esta forma se garantiza que no se generen ejemplos sintéticos en zonas, que lejos de mejorar, empeoren el comportamiento de los clasificadores. Demostrar la viabilidad de la propuesta a través de análisis comparativo con varios algoritmos del estado del arte.

- Proponer un nuevo algoritmo híbrido de preprocesamiento para conjuntos de alto desbalance que utilice SMOTE para generar nuevos ejemplos y evalúe la pertenencia de los ejemplos sintéticos y los mayoritarios originales a la región positiva de su clase, utilizando el enfoque difuso de la TCA. El objetivo de esta propuesta es que en conjunto final solo queden aquellas instancias sintéticas y mayoritarias originales cuyo grado de pertenencia a la región positiva de su clase esté por encima de un

umbral dado, considerando el concepto de similaridad difusa para evaluar la calidad de los ejemplos de la clase mayoritaria y los sintéticos. Demostrar la viabilidad de la propuesta a través de análisis comparativo con varios algoritmos del estado del arte.

- Proponer un nuevo algoritmo híbrido de preprocesamiento para conjuntos de alto desbalance que utilice SMOTE y el enfoque difuso de la TCA con doble umbral y aplicarlo al problema del diagnóstico de la necesidad de mantenimiento de los interruptores de alta potencia. El objetivo es diseñar un algoritmo que se ajuste a las necesidades reales de una aplicación en la ingeniería, por tanto se deben crear ejemplos sintéticos de alta pertenencia a la clase minoritaria (usar un umbral alto) y eliminar ejemplos originales mayoritarios que tengan muy baja pertenencia a la región positiva de su clase (usar un umbral muy bajo).
- Proponer un nuevo algoritmo de clasificación para conjuntos de datos no balanceados usando la TCAD y la agregación con el operador OWA utilizando vectores de pesos para ponderar los ejemplos. El objetivo de esta propuesta es lograr un algoritmo que durante el aprendizaje construya vectores de pesos atendiendo a la representatividad de cada clase y usar ese vector para determinar el grado de pertenencia a la región positiva de cada clase de los ejemplos a clasificar. Demostrar la viabilidad de la propuesta a través de análisis comparativo con varios algoritmos del estado del arte.
- Para el cumplimiento de los objetivos trazados en la tesis, se desarrolló un estudio previo de los mejores algoritmos existentes en el estado del arte para el tratamiento de datos no balanceados. Además se estudió la TCA y su enfoque difuso, como potentes herramientas matemáticas para discernir los ejemplos que son mejores representantes de sus clases, a través del concepto de aproximación inferior.

Partiendo de este estudio se diseñaron algoritmos al nivel de los datos los cuales primeramente igualaban la cantidad de ejemplos en cada clase usando SMOTE y luego evaluaban la pertenencia de los objetos a su clase usando la TCA y la TCAD. Uno de los algoritmos propuestos se utilizó en la solución de un problema real: el diagnóstico de la necesidad de mantenimiento de los interruptores de potencia. Los algoritmos creados han demostrado un comportamiento robusto frente a datos altamente desbalanceados (IR>9), el uso de la TCA y la TCAD ha permitido que se inserten el conjunto final solo aquellos

ejemplos que pertenecen con un alto grado a su clase, evitando la presencia de ejemplos en la frontera.

A partir del concepto de grado de pertenencia a la región positiva, dado por la TCAD, y la agregación con el operador OWA utilizando vectores de pesos para ponderar los ejemplos, se propuso un nuevo algoritmo de clasificación para conjuntos no balanceados partiendo del algoritmo Fuzzy Rough Nearest Neighbor (FRNN). Se diseñan 6 estrategias diferentes para la creación de los vectores de pesos. Este método logra excelentes resultados sobre conjuntos con diferentes índices de desbalance (desde 1.8 hasta 129) logrando resultados significativamente superiores a reconocidos algoritmos diseñados para la clasificación no balanceada y a varios algoritmos de preprocesamiento combinados con potentes clasificadores como kNN, C4.5 y SVM.

Summary

In this thesis we introduce new approaches for deal with imbalanced data. We used the Rough Set Theory and the Fuzzy Rough Set Theory to evaluate the instances membership to the positive region. We focus our proposals in two levels: data level, when we proposed hybrids method for preprocessing imbalanced datasets, and algorithms level, when we proposed new method to classified imbalanced data.

Our objectives are the following:

- To propose a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets, using SMOTE to generate artificial minority examples and Rough Sets Theory to evaluate the created examples. The main objective of this proposal is to guarantee that every synthetic example created by SMOTE belong to the lower approximation of its class, avoiding the introduction of boundaries examples. To validate the proposed algorithm thought an experimental study.
- To propose a hybrid preprocessing approach for high imbalanced data-sets, using SMOTE to generate artificial minority examples and Fuzzy Rough Sets Theory to evaluate the created and the majorities examples. The objective is to eliminate all

synthetic examples or majorities' ones that do not belong to the positive region with a value higher than a threshold. To demonstrate the goodness of the proposed algorithm thought an experimental study.

- To propose a hybrid preprocessing approach for high imbalanced data-sets, using SMOTE to generate artificial minority examples and Fuzzy Rough Sets Theory to evaluate the created and the majorities examples, using two different thresholds. Applied the resulting algorithm to solve the problem of the diagnosis of high voltage circuit breaker maintenance. The objective is to eliminate all synthetic examples or majorities' ones that do not belong to the positive in a certain degree. We proposed the use of a very high value for the threshold to eliminate synthetic examples in order to select only the best, and the use of a very small value for the original majorities in order to only remove the worst. To demonstrate the goodness of the proposed algorithm thought an experimental study.
- To propose a classification algorithm for imbalanced data that uses Fuzzy Rough Set Theory and ordered weighted average aggregation. This proposal for every example to be classified in the learning constructs the weigh vectors according to the representativeness of the class and the membership degree to the positive region. To demonstrate the goodness of the proposed algorithm thought an experimental study. To carry out the bellow objectives, we develop a preliminary study of the state-of-the-art methods design for dealing with imbalanced data-sets.

We study the Rough Set theory and the Fuzzy Rough Set Theory as powerful tools for determinate the best representatives examples in a training set, trough the concept of lower approximation. We propose a set of preprocessing approaches for high imbalanced data-set using SMOTE for making the distribution between classes in imbalanced training sets uniform and RST and FRST to evaluate the created or the majorities examples. The created algorithms demonstrated a robustness behavior front data-set with an imbalanced ratio higher than 9.

We proposed a classification algorithm based on the algorithm Fuzzy Rough Nearest Neighbor (FRNN) for imbalanced data that uses FRST and ordered weighted average aggregation (OWA). The proposal considers different strategies to build a weight vector for taking into account data imbalance. The statistical analysis have shown that IFROWANN

can outperform not only the classical FRNN algorithm over a large collection of imbalanced data sets with varying IR degrees, but also a selection of representative algorithms from the state-of-the-art that cover data-level (combined with C4.5, kNN and SVM), cost-sensitive and ensemble solutions specifically designed for imbalanced learning.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

Índice de algoritmos

LISTA DE ABREVIATURAS

AA Aprendizaje automático

ANSI Instituto Americano de Normas Nacionales

AUC Área bajo la curva ROC

CIGRE Sistemas Eléctricos de Alta Tensión por sus siglas en francés

EUS Bajo muestreo evolutivo (Evolutionary UnderSampling)

FRNN Vecino más cercano usando los conjuntos aproximados difusos (Fuzzy Rough Nearest Neighbor)

HVCB High Voltage Circuit Breaker (IAP en español)

IAP Interruptor de Alta Potencia

IEEE Instituto de Ingenieros Eléctricos y Electrónicos

IR Índice de desbalance (Imbalanced Ratio)

kA Kilo ampere

KEEL Extracción de conocimiento y aprendizaje evolutivo (Knowledge Extraction and Evolutionary Learning)

NCL Regla de limpieza basada en vecindad (Neighborhood Cleaning Rule)

OWA Promedio ponderado ordenado (Ordered Weighted Average)

ROC Característica Operativa del Receptor (Receiver Operating Characteristic)

SMOTE Técnica para sobremuestraer con ejemplos sintéticos la clase minoritaria (Synthetic Minority Oversampling Technique)

TCA Teoría de los Conjuntos Aproximados (Rough Set Theory)

TCAD Teoría de los Conjuntos Aproximados Difusos

tra Training, conjunto de entrenamiento

tst test, conjunto de prueba

VCCAD Vecinos más Cercanos con Conjuntos Aproximados Difusos

Planteamiento

"A little knowledge that acts is worth infinitely more than much knowledge that is idle."
– Kahlil Gibran.

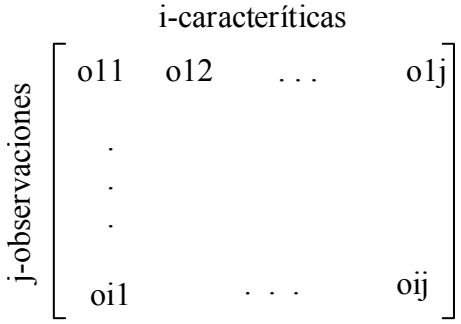
Motivación

La creación de herramientas informáticas que "aprendan" a partir de la "experiencia" y sean capaces de diagnosticar, recomendar, predecir un hecho del que no existe precedente, o es similar a uno ocurrido; desde hace algunos años atrapa la atención de la comunidad científica del área de la computación.

La necesidad de disponer de algoritmos que permitan de forma automática la generación de modelos a partir de los datos disponibles, ha propiciado el desarrollo vertiginoso de la minería de datos J.Han (2005); Tan et al. (2005); Wang and Fu (2005), y dentro de ésta del aprendizaje automático (AA)Cherkassky and Mulier (2007); Mitchell (1997) como área que estudia la extracción del conocimiento implícito en los datosWitten and Frank (2005).

Para llevar a cabo un proceso de AA, se necesita en primer lugar de un conjunto de datos de los que aprenderá el clasificador. Un conjunto de datos es una colección de observaciones cuidadosamente recogidas de algún suceso en estudio, descritas por características. Su estructura es una matriz donde cada fila representa una observación y cada columna una característica de la observación, como se muestra en la figura 1. Si a cada fila (hecho) se le adiciona una nueva característica que represente su categoría o clasificación, entonces el conjunto de datos estará preparado para ser sometido a un proceso de clasificación. Si

Figura 1: Conjunto de datos



dicha característica es de naturaleza continua se denomina regresión, si por el contrario es discreta se denomina clasificación.

La clasificación es el proceso a través del cual se le asigna a un nuevo objeto, una clase o categoría atendiendo a sus características. Por ejemplo: conociendo todos los síntomas de un paciente del que se desconoce su enfermedad, asignarle una "enfermedad". Sin embargo los datos pueden contener ciertas características que afecten el proceso de aprendizaje, y que requieran por tanto de un paso previo antes del mismo, a este paso intermedio se le conoce como *preprocesamiento*.

Una de las características que pueden hacer necesario un preprocesamiento es la conocida como desbalance de clases Chawla et al. (2004a); He and García (2009); Sun et al. (2009). Un conjunto de datos es balanceado si tiene aproximadamente igual porcentaje de ejemplos en los conceptos a clasificar, es decir, si la distribución de ejemplos por clases es uniforme. Sin embargo, en la realidad los sucesos casi nunca ocurren con la misma frecuencia, sin mencionar aquellos sucesos "raros" que ocurren esporádicamente J.H.Zhao et al. (2007), por lo que al captar los datos suelen aparecer muchos ejemplos de una clase (que sería el estado normal) y pocos de la otra (que sería la anomalía o suceso poco frecuente). Dada que cada vez son más las aplicaciones que se enfrentan a este problema Cohen et al. (2006); Huang et al. (2006); J.H.Zhao et al. (2007); Khreich et al. (2010); Lee et al. (2013), el mismo ha venido cobrando auge hasta convertirse en uno de los problemas de aprendizaje que han centrado la atención de los investigadores del área de AA en los últimos años.

El aprendizaje a partir de datos no balanceados es un problema que afecta el comportamiento de los algoritmos de aprendizaje. La tendencia de los clasificadores es a ignorar la distribución por clases y a considerar los ejemplos de la clase menos representada como ruido. Como consecuencia los ejemplos de la clase menos representada tienden a estar mal clasificados y en la práctica éstos suelen ser los de mayor interés.

Para hacer frente al problema del desbalance de clases han surgido un grupo de técnicas dividas en 4 niveles: al nivel de los datos, al nivel de los algoritmos de aprendizaje, las del aprendizaje sensible al coste y las basadas en multiclasificadores. Las técnicas al nivel de los datos son hasta al momento las más usadas, debido a que su uso es independiente al clasificador que se seleccione.

Dentro de las técnicas al nivel de los datos, las que mejores resultados logran son las conocidas como híbridas, pues las mismas generan ejemplos sintéticos de la clase menos representada y luego aplican un algoritmo de limpieza, el cual atendiendo a cierto criterio elimina los ejemplos que no considere "buenos" para la etapa de aprendizaje.

Uno de los algoritmos más conocidos dentro de las técnicas al nivel de los datos es el algoritmo de generación de ejemplos sintéticos SMOTE (Synthetic Minority Oversampling Technique) Chawla et al. (2002). Dicho algoritmo para cada ejemplo de la clase menos representada introduce ejemplos sintéticos entre él y sus vecinos más cercados. SMOTE tiene la desventaja que puede introducir ejemplos en el área de la clase más representada, es por ello que varios autores lo han combinado con algoritmos de limpieza, creando nuevos algoritmos híbridos que mejoran los resultados del SMOTE original.

Sin embargo, y a pesar de existir ya un número considerable de algoritmos híbridos para preprocesar conjuntos de entrenamiento no balanceados Batista et al. (2004); Bunkhumpornpat et al. (2009); Han et al. (2005); Napierala et al. (2010), éstos no son capaces de generar ejemplos sintéticos y lograr a su vez que estos ejemplos pertenezcan con certeza a su clase, de esta forma se generan muchos ejemplos en el área de la clase más representada y esto afecta el comportamiento de los clasificadores.

Algunos de los algoritmos de preprocesamiento existentes en el estado del arte aplican los algoritmos de limpieza sobre todo el conjunto resultante (originales + sintéticos), evaluando

a todos con el mismo rigor. Esto puede provocar la pérdida excesiva de ejemplos originales valiosos para la etapa de aprendizaje. No siempre es provechoso aplicar los criterios de limpieza con el mismo rigor sobre sintéticos y originales. Resulta obvio que mientras más estricto sea el criterio de evaluación de los ejemplos sintéticos más calidad tendrán los mismos. Por otra parte, puede que dentro de los ejemplos originales existan ejemplos ruidosos que no sean buenos representantes de su clase y que su eliminación sea beneficiosa para la etapa de aprendizaje, pero, eliminar estos ejemplos usando una condición muy estricta, y teniendo en cuenta la baja representatividad de una de las clases, puede llevarnos a perder demasiados ejemplos.

La Teoría de los Conjuntos Aproximados (TCA) Pawlak (1982, 1995); Pawlak et al. (1995); S.K.M. et al. (1988), es una poderosa herramienta para el análisis de datos. La misma es considerada como una de las cinco áreas claves y no tradicionales de la Inteligencia Artificial y de la Teoría de la Información Incompleta, pues constituye una herramienta muy útil para el manejo de la información no completa o imprecisa. En la TCA clásica la separabilidad es vista como un modelo en blanco y negro (las instancias son separables o no), lo cual limita en cierta forma la aplicación de la teoría. Esta dificultad ha dado lugar al surgimiento la Teoría de los Conjuntos Aproximados Difusa (TCAD) en la cual las relaciones de inseparabilidad son modeladas por medio de relaciones difusas simétricas y reflexivas.

La TCA y la TCAD proveen tres conceptos fundamentales que permiten categorizar los ejemplos en un sistema de decisión atendiendo a sí son buenos representantes o no de su clase. Estos conceptos han sido ampliamente utilizados en el preprocesamiento de datos y en sistemas de clasificación.

Por todo lo planteado se formula como **problema científico** que:

Los algoritmos de generación de instancias artificiales para conjuntos no balanceados, no garantizan que los ejemplos sintéticos generados pertenezcan con un alto grado de certeza a su clase; además los algoritmos de limpieza que se aplican utilizan los mismos criterios de evaluación para los ejemplos originales y sintéticos, lo cual puede provocar la eliminación de ejemplos originales valiosos para la etapa de aprendizaje. Por otra parte, la

mayoría de los algoritmos de clasificación tradicionales no tienen en cuenta la distribución de ejemplos por clases a la hora de clasificar.

A partir de problema científico identificado nos planteamos como **hipótesis** que:

Hipótesis 1: La utilización del concepto de aproximación inferior de la Teoría de los Conjuntos Aproximados, tanto la clásica como difusa, para evaluar la calidad de los ejemplos en conjuntos de entrenamiento no balanceados, logra mejorar el comportamiento de los clasificadores frente a conjuntos de datos no balanceados.

Hipótesis 2: La determinación del grado de pertenencia de un objeto a cada clase, valiéndonos del uso de los conceptos de región positiva y negativa dado por la Teoría de los Conjuntos Aproximados Difusa y los operadores de agregación, mejora de manera significativa la clasificación de datos no balanceados.

Objetivos

A partir del problema científico identificado se define como objetivo general de la investigación: *Diseñar algoritmos de preprocesamiento híbridos que evalúen la calidad de los ejemplos a través de la pertenencia a la región positiva de su clase valiéndonos de la Teoría de los Conjuntos Aproximados tanto clásica como difusa y usando criterios de edición menos restrictivos o nulos para los ejemplos originales; así como diseñar algoritmos de clasificación que utilicen la pertenencia de los ejemplos a la región positiva difusa de su clase evaluando la importancia de los ejemplos aprendidos según la representatividad de la clase a la que pertenecen con el uso de los correspondientes operadores de agregación.*

Para cumplir el objetivo general, lo hemos desglosado en los siguientes objetivos específicos:

- *Proponer un nuevo algoritmo híbrido de preprocesamiento para conjuntos de alto desbalance que utilice SMOTE para generar nuevos ejemplos y evalúe la calidad*

de dichos ejemplos a través de la TCA. El objetivo de esta propuesta es que cada instancia sintética que sea generada con SMOTE pertenezca a la aproximación inferior de la clase, de esta forma se garantiza que no se generen ejemplos sintéticos en zonas, que lejos de mejorar, empeoren el comportamiento de los clasificadores. Demostrar la viabilidad de la propuesta a través de análisis comparativo con varios algoritmos del estado del arte.

- *Proponer un nuevo algoritmo híbrido de preprocesamiento para conjuntos de alto desbalance que utilice SMOTE para generar nuevos ejemplos y evalúe la pertenencia de los ejemplos sintéticos y los mayoritarios originales a la región positiva de su clase, utilizando el enfoque difuso de la TCA*. El objetivo de esta propuesta es que en conjunto final solo queden aquellas instancias sintéticas y mayoritarias originales cuyo grado de pertenencia a la región positiva de su clase esté por encima de un umbral dado, considerando el concepto de similaridad difusa para evaluar la calidad de los ejemplos de la clase mayoritaria y los sintéticos. Demostrar la viabilidad de la propuesta a través de análisis comparativo con varios algoritmos del estado del arte.
- *Proponer un nuevo algoritmo híbrido de preprocesamiento para conjuntos de alto desbalance que utilice SMOTE y el enfoque difuso de la TCA con doble umbral y aplicarlo al problema del diagnóstico de la necesidad de mantenimiento de los interruptores de alta potencia*. El objetivo es diseñar un algoritmo que se ajuste a las necesidades reales de una aplicación en la ingeniería, por tanto se deben crear ejemplos sintéticos de alta pertenencia a la clase minoritaria (usar un umbral alto) y eliminar ejemplos originales mayoritarios que tengan muy baja pertenencia a la región positiva de su clase (usar un umbral muy bajo).
- *Proponer un nuevo algoritmo de clasificación para conjuntos de datos no balanceados usando la TCAD y la agregación con el operador OWA utilizando vectores de pesos para ponderar los ejemplos*. El objetivo de esta propuesta es lograr un algoritmo que durante el aprendizaje construya vectores de pesos atendiendo a la representatividad de cada clase y usar ese vector para determinar el grado de pertenencia a la región positiva de cada clase de los ejemplos a clasificar. Demostrar la viabilidad de la propuesta a través de análisis comparativo con varios algoritmos del estado del arte.

Estructura de la memoria

Esta memoria está compuesta por el presente Planteamiento, en la cual se define la situación problémica que da origen a nuestra tesis así como los objetivos que nos hemos trazado. La memoria consta además de otros 4 capítulos en los que se detalla la investigación realizada.

En capítulo 1, se hace un estudio del marco teórico relacionado con el aprendizaje a partir de datos no balanceados, los principales algoritmos del estado del arte y su clasificación así como la evaluación en dominios no balanceados. Se describen los elementos teóricos relacionados con la teoría de los conjuntos aproximados, tanto la clásica como la difusa, así como sus aplicaciones en el preprocesamiento.

En el capítulo 2 se propone un nuevo algoritmo de preprocesamiento llamado SMOTE-RSB* el cual utiliza el algoritmo SMOTE para generar ejemplos sintéticos y luego aplica un algoritmo de limpieza sobre los ejemplos sintéticos basado en la Teoría de los Conjuntos Aproximados.

En el capítulo 3 se propone un nuevo algoritmo de preprocesamiento llamado SMOTE-TCAD basado en SMOTE para generar ejemplos sintéticos luego aplica un algoritmo de limpieza a los ejemplos sintéticos y los mayoritarios originales basados en la Teoría de los Conjuntos Aproximados Difusa. En un segundo momento se propone una extensión de SMOTE-TCAD llamada SMOTE-TCAD-2T la cual edita de forma diferente los ejemplos sintéticos y los de la clase mayoritaria originales, dejando al igual que el primero los ejemplos de la clase minoritaria originales sin tocar. En la sección 3.4 se presenta una aplicación del algoritmo SMOTE-TCAD-2T, siendo usado para el diagnóstico de la necesidad de mantenimiento de los interruptores de alta potencia.

En el capítulo 4 se presenta un nuevo algoritmo de clasificación para conjuntos no balanceados usando el algoritmo Aproximación Difusa del Vecino más Cercano (Vecinos más Cercanos con Conjuntos Aproximados Difusos), se proponen 6 nuevas estrategias de ponderación para la agregación con el operador OWA.

En el último capítulo, se resumen los principales resultados y aportes de la investigación, comentando los mismos y planteando algunos de los trabajos futuros que se pueden abordar en el área.

CAPÍTULO 1

Introducción: Aprendizaje Automático y la Teoría de Conjuntos Aproximados

"Nothing in life is to be feared. It is only to be understood."
– Marie Currie.

En este capítulo se presentan los elementos teóricos estudiados en la realización de la presente tesis. Primeramente se hace una breve introducción al aprendizaje automático, para centrarnos posteriormente en el problema de clasificación y en el caso concreto de la clasificación no balanceada; se describen las principales técnicas para abordar el problema. En un segundo momento se enuncian los principales conceptos de la TCA y su enfoque difuso. Se presenta además un estudio de la utilización de la TCA y TCAD en el preprocesamiento de datos, así como su aplicación en dominios no balanceados.

1.1. Aprendizaje automático

La cantidad de datos que acumula la actividad del hombre es incontable. Millones de tuplas son registradas a diario en las bases de datos, cada una de ellas constituye una observación,

una experiencia de la que aprender, una situación que pudiera volver a ocurrir en el futuro de forma similar. Aprender a partir de la experiencia es algo que los humanos hacemos de forma natural y constantemente, pero ¿qué ocurre si el número de "experiencias" excede nuestra capacidad para procesarla?, ¿qué ocurre si un hecho se repite millones de veces y nunca vuelve a ocurrir exactamente de la misma forma?.

El aprendizaje automático es el área de la Inteligencia Artificial, que se ocupa de "aprender" a partir de la "experiencia", es decir, extraer de forma automática conocimiento implícito en la información (almacenada en forma de datos) Mitchell (1997). El mismo ha cobrado un gran auge en los últimos años debido a su gran aplicabilidad en problemas reales Mazurowskia et al. (2008); Merschmann and Plastino (2007); Peng and King (2008).

Dentro del aprendizaje automático se encuentra el aprendizaje inductivo, este tipo de método asume que se conoce un conjunto de ejemplos o instancias Kasabov (1998). Formalnente se define como:

Definición 1 *Sean las instancias de la forma* (x_i, y_i)*,* $x_i \in D$ *es un estado del espacio de dominio D y* $y_i \in S$ *es un estado del espacio de dominio de solución S, o la forma* (x_i)*,* $i = 1, 2, ..., n$*, donde no se especifiquen los vectores de salida. La tarea de crear un sistema que pueda aprender las parejas de entrada-salida* $\{(x, y)\}$*o aprender las características inherentes a* $\{x\}$ *se define como Aprendizaje.*

El primero de los casos se refiere al *aprendizaje supervisado*, donde se tiene una solución y_i (la etiqueta de la clase) para cada vector x_i de entrada, a estos ejemplos se les conoce como "clasificados" o "etiquetados" Cherkassky and Mulier (2007). El segundo caso se refiere al *aprendizaje no supervisado*, el cual consiste en que un sistema aprende características, rasgos, grupos, conceptos a partir de datos no etiquetados.

1.2. Introducción al problema de clasificación

La clasificación consiste en el proceso de asignar a un objeto específico, el nombre de la clase a la que pertenece (aprendizaje supervisado) Wang and Fu (2005). Las clases resultan

de un problema de predicción, donde cada clase corresponde a la salida posible de una función a predecir a partir de los atributos con los que describimos los elementos de la base de datos Han and Kamber (2000); Mitchell (1997). La necesidad de un clasificador surge por requerimientos de disponer de un procedimiento mecánico mucho más rápido que un supervisor humano y que a la vez pueda evitar sesgos y prejuicios adoptados por un experto Mitra and Acharya (2003). Formalmente se define como:

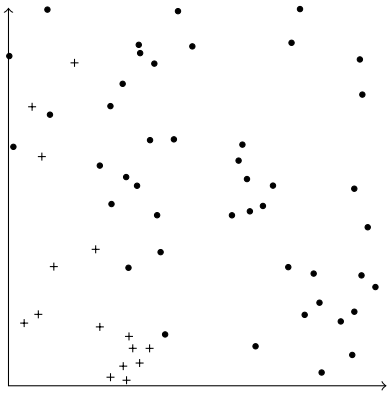
Sea T un conjunto de entrenamiento, el cual está formado por parejas (x_i, y_i),$i = 1, ..., n$ donde n representa la cantidad de atributos, x_i define un vector de entrada mientras y_i define su etiqueta clase (salida). A la tarea de asignar una salida y_i a una entrada x_i se le conoce como Clasificación.

Definición 2 *Sea T un conjunto de entrenamiento, el cual está formado por parejas (x_i, y_i),$i = 1, ..., n$ donde n representa la cantidad de atributos, x_i define un vector de entrada mientras y_i define su etiqueta clase (salida). A la tarea de asignar una salida y_i a una entrada x_i se le conoce como Clasificación.*

Para llevar a cabo un proceso de clasificación se requiere de un mecanismo capaz de crear un modelo a partir del conjunto de entrenamiento, a este mecanismo se conoce como "clasificador" y existen infinidad de éstos de operan de dfiferente maneras, dentro de los más conocidos se encuentran las redes neuronales Bishop (1995); McCulloch and Pitts (1943), los árboles de decisión Quinlan (1988, 1993), máquinas de soporte vectorial Schölkopf and Smola. (2001); Vapnik (1995), clasificadores probabilísticos John and Langley (1995) y los basados en vecindad Cover and Hart (1967).

Los conjuntos de datos en su estado original (tal y como son captados) pueden traer algunas características que afecten el proceso de clasificación como pudieran ser: presencia de ruido, de valores ausentes, exceso de características que afectan la separabilidad de clases, ejemplos repetidos, ejemplos que no son una buena representación de su clase, baja representatividad de uno de los conceptos a clasificar, entre otros. Es por ello que puede ser necesaria una etapa previa conocida como *preprocesamiento*. La preparación o *preprocesamiento* de datos engloba a todas aquellas técnicas de análisis

Figura 1.1: Ejemplo de un conjunto no balanceado



de datos que permiten mejorar la calidad de un conjunto de ellos, de modo que los algoritmos de extracción de conocimiento puedan obtener mayor y mejor información (mejor comportamiento en la clasificación supervisada, reglas con más completitud, entre otros) Zhang and Zhang (2003).

Uno de los problemas que con más frecuencia suele aparecer en los datos es la baja representatividad de uno de los conceptos a clasificar, en el siguiente epígrafe nos referiremos a este problema.

1.3. Clasificación a partir de datos no balanceados

Actualmente uno de los grandes desafíos a los que se enfrenta la minería de datos es el aprendizaje a partir de datos no balanceados Yang and Wu (2006). La ocurrencia de sucesos poco frecuentes ha dado lugar a que exista una desproporción considerable entre el número de ejemplos en cada clase, la clase menos representada (positiva o minoritaria) suele ser la de mayor interés en el problema de clasificación, mientras que la más representada (negativa o mayoritaria) constituye simplemente contraejemplos de la positiva.

La figura 1.1 muestra un ejemplo de un conjunto no balanceado, como se puede observar la clase positiva además de estar poco representada también está dispersa y mezclada con la negativa lo que constituye otra agravante para la etapa de aprendizaje.

En la práctica no todos los sucesos ocurren con la misma frecuencia, y si la poca representatividad de uno de los conceptos no es tenido en cuenta a la hora de resolver el problema mediante técnicas de la minería de datos, se pueden obtener resultados erróneos, debido a que los resultados globales no son afectados por la desigual distribución de ejemplos por clases.

Existen numerosas aplicaciones del aprendizaje automático donde se encuentra este problema, como por ejemplo el diagnóstico de enfermedades con condiciones médicas poco frecuentes Mazurowski et al. (2008), detección en línea de eventos raros J.H.Zhao et al. (2007), en la bioinformática L.M.Taft et al. (2009), en el análisis de fallas de transformadores de energía Yang et al. (2009), detección de fraude Tavallaee et al. (2010), entre otras.

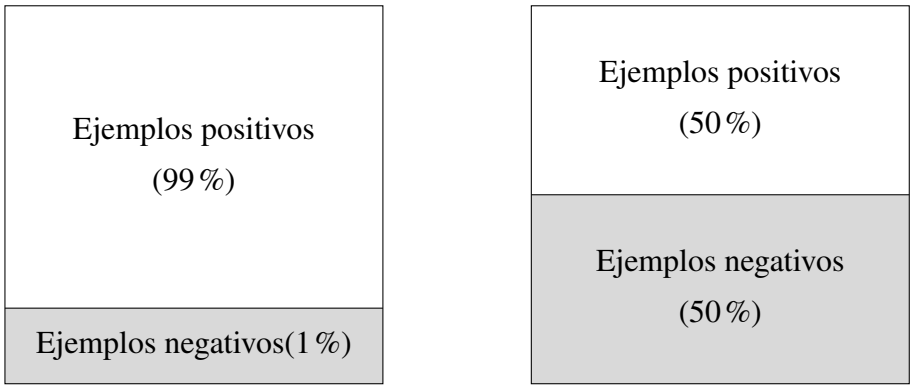
Los algoritmos de clasificación tienen tendencia a clasificar muy bien los ejemplos de la clase negativa, mientras que los de la clase positiva tienden a estar mal clasificados. Esto ocurre dado muchos clasificadores consideran los datos menos frecuentes como rarezas o ruido, centrándose únicamente en métricas globales que no tienen en cuenta la distribución de ejemplos por clases He and García (2009); Orriols-Puig and Bernado-Mansilla (2009); Sun et al. (2009).

La figura 1.2 muestra un ejemplo de un conjunto cuya clase minoritaria representa solo el $1\,\%$ del total de ejemplos (a), en (b) se muestra un conjunto donde cada una de las clases representa exactamente la mitad del conjunto. Si este conjunto es sometido a un proceso de clasificación, donde el clasificador acierta en todas las instancias de la clase mayoritaria y falla en todas las de la clase minoritaria, la exactitud general obtenida será del $99\,\%$, como se observa en la figura 1.3, es por ello que es necesario usar otro tipo de métricas para problemas desbalanceados, esta problemática será abordada en detalles en siguientes epígrafes.

Numerosas técnicas han surgido para lidiar con el problema del desbalance de clases, estas técnicas están divididas en 3 grandes grupos:

- **Al nivel de los datos** (preprocesamiento) Batista et al. (2004); Chawla et al. (2002, 2008); López et al. (2012): estás técnicas conocidas como remuestreo se centran en

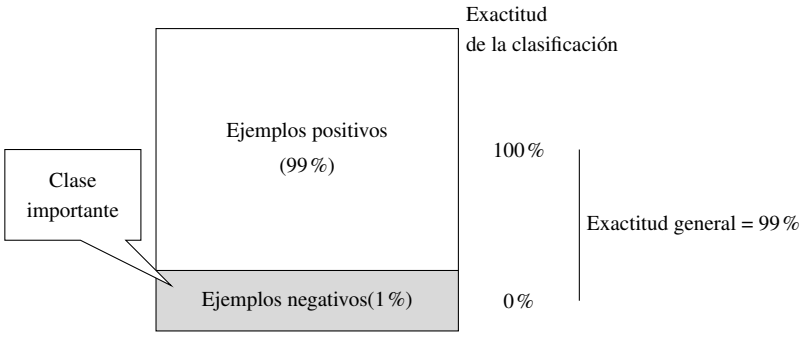
Figura 1.2: Datos desbalanceados y balanceados



(a)
Datos desbalanceados

(b)
Datos balanceados

Figura 1.3: El problema de los clasificadores frente a conjuntos no balanceados



intentar balancear el conjunto, es decir, tratan que las clases queden igualadas en cuanto al número de ejemplos por clases. Pueden ser de bajo-muestreo: que eliminan ejemplos de la clase más representada; de sobre-muestreo: que generan ejemplos sintéticos de la clase menos representada, o híbridas que combinan ambas técnicas anteriores.

- **Al nivel de los algoritmos** Huang et al. (2006): estas técnicas consisten en el diseño de algoritmos específicos que tienen en cuenta la desproporción de ejemplos por clases, por ejemplo ajustando la estimación de probabilidad en las hojas de un árbol de decisión a favor de la clase positiva Weiss and Provost (2003).
- **Aprendizaje sensible al coste** Domingos (1999); López et al. (2012); Ting (2002); Zadrozny et al. (2003); Zhou and Liu (2010): Este tipo de algoritmos incorpora soluciones al nivel de los datos, al nivel de los algoritmos, o a los dos niveles simultáneamente, asignando un mayor coste al error de fallar en un ejemplo positivo que en uno negativo, y de esta forma minimizar el mayor error de coste.
- **Soluciones con multiclasificadores** Galar et al. (2012): Las técnicas multiclasificadores para clasificación con datos no balanceados consisten usualmente en una combinación de un algoritmo de aprendizaje ensemble con alguna de las técnicas descritas anteriormente, específicamente las técnicas al nivel de los datos y las que usan aprendizaje sensible al coste. Cuando se combina un ensemble con una técnica al nivel de los datos, el nuevo algoritmo híbrido usualmente preprocesa el conjunto antes del entrenar cada clasificador.

La gran ventaja de las técnicas al nivel de los datos que son más versátiles desde el punto de vista que su uso es independiente del clasificador que se seleccione. En esta investigación se estudian varios algoritmos de selección de instancias combinados con algoritmos de sobre-muestreo y técnicas híbridas para ajustar la distribución por clases en un conjunto de entrenamiento. Específicamente se han seleccionado los algoritmos estudiados en Batista et al. (2004), estos algoritmos están clasificados en los siguientes tres grupos:

- **algoritmos de bajo-muestreo**: Crean un subconjunto del conjunto original a través de la eliminación de algunos ejemplos de la clase mayoritaria.

- **algoritmos de sobre-muestreo**: Crean un superconjunto del conjunto original, replicando algunos ejemplos de la clase minoritaria o creando nuevos a partir de los originales.
- **algoritmos híbridos**: Combinan las dos técnicas anteriores, eliminando algunos de los ejemplos minoritarios que fueron introducidos con el sobre-muestreo, con el objetivo de eliminar el sobreaprendizaje.

1.3.1. Algoritmos al nivel de los datos

En esta subsección se describirán en detalles los algoritmos al nivel de los datos más reconocidos del estado del arte.

Dentro de las técnicas de bajo-muestro más usadas se encuentran:

- "Tomeklinks" Tomek (1976), definido como: dados dos ejemplos e_i y e_j que pertenecen a clases distintas y $d(e_i, e_i)$ es la distancia entre los ejemplos e_i y e_j. El par (e_i, e_j) es llamado TomekLink si no existe un ejemplo E_l , tal que $d(e_i, e_l) < d(e_i, d_j)$ o $d(e_j, e_l) < d(e_i, e_j)$. Si dos ejemplos forman un par Tomek link, es porque al menos uno de estos dos ejemplos es ruido o los 2 son bordes. Este algoritmo puedes ser usado como bajo-muestreo o como algoritmo de limpieza, como algoritmo de bajo-muestreo solo ejemplos de la clase mayoritaria son eliminados pero como algoritmo de limpieza se eliminan de las dos clases.
- "Neighborhood Cleaning Rule" (NCL): este algoritmo usa la Edición del Vecino más Cercano (ENN por sus siglas en inglés) de Wilson Wilson (1972), para eliminar ejemplos de la clase mayoritaria. ENN elimina cualquier ejemplo que su etiqueta de clase sea diferente a la clase de al menos dos de sus tres vecinos más cercanos. NCL modifica a ENN con el objetivo de aumentar la limpieza de datos. Para un problema de 2 clases el algoritmo funciona de la siguiente forma: para para ejemplo e_i en el conjunto de entrenamiento, se buscan sus tres vecinos más cercanos. Si e_i pertenece a la clase mayoritaria y la clasificación dada por sus tres vecinos más cercanos contradice la clase original de e_i, entonces e_i es eliminado. Si e_i pertenece

a la clase minoritaria y sus tres vecinos más cercanos lo clasifican mal, entonces se eliminan los vecinos que pertenezcan a la clase mayoritaria.

El algoritmo SMOTE Chawla et al. (2002) es uno de los más reconocidos dentro de los algoritmo de sobre-muestreo:

- "Synthetic Minority Oversampling Technique" (SMOTE): este algoritmo crea nuevos ejemplos sintéticos de la clase minoritaria a través de la interpolación entre algunos ejemplos de la clase minoritaria con sus respectivos vecinos más cercanos. Un problema que presenta SMOTE es que puede crear ejemplos sintéticos en el área de la clase mayoritaria.

Los algoritmo híbridos basados en SMOTE, logran un buen equilibrio entre introducir ejemplos nuevos de la clase minoritaria y bajo algún criterio eliminar algunos de ellos, este grupo de algoritmo son los que reportan mejores resultados.

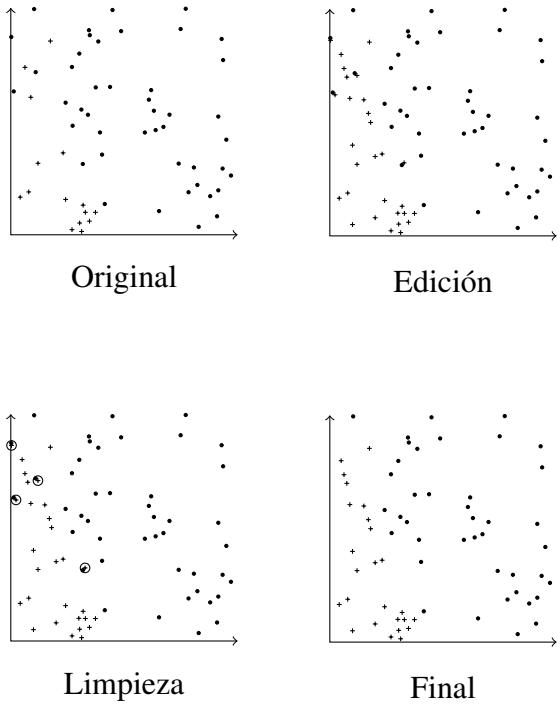
- "SMOTE - Tomeklinks": Con frecuencia, los clúster de clases no están bien definidos pues algunos ejemplos de la clase mayoritaria pueden invadir el área de la clase minoritaria, lo contrario también puede ocurrir, pues la interpolación que se realiza entre los ejemplos de la clase minoritaria, puede introducir ejemplos minoritarios en el área de la clase mayoritaria. Inducir un clasificador en estas circunstancias puede llevarnos al overfitting. Con el objetivo de definir mejor los clúster de clases Batista y colaboradores Batista et al. (2004) proponen un nuevo algoritmo para aplicar Tomek link como algoritmo de limpieza sobre los conjuntos sobre-muestrados con SMOTE. Este algoritmo elimina ejemplos de las 2 clases. La figura 1.4 muestra en detalles el comportamiento de este algoritmo.
- "SMOTE - ENN": este algoritmo es similar al anterior Batista et al. (2004). ENN elimina más ejemplos que TomekLink, es un algoritmo de limpieza más potente. Al contrario de NCL que es solo un algoritmo de bajo-muestreo ENN es usado para eliminar ejemplos de ambas clases, de esta forma cualquier ejemplo que sea mal clasificado por sus tres vecinos más cercanos será eliminado del conjunto de entrenamiento.

- "Borderline-SMOTE1": este algoritmo solo genera ejemplos los ejemplos minoritarios en el área del borde la clase (borderline examples), Han et al. (2005). Primero busca los ejemplos minoritarios en el borde P, luego se generan ejemplos sintéticos que son agregados al conjunto de entrenamiento original. Este algoritmo cada ejemplo minoritario (p_i) calcula sus m vecinos más cercanos en todo el conjunto de entrenamiento, si todos los m vecinos más cercanos son de la clase mayoritaria, el ejemplo p_i es considerado ruido y no pasa el próximo paso. Si $m/2 \leq m' < m$, p_i es puesto en el conjunto $DANGER$. Si $0 \leq m' < m/2$, p_i es seguro y no necesita pasar al siguiente paso. Los ejemplos en $DANGER$ son los ejemplos bordes de la clase minoritaria P. Finalmente a cada elemento puesto en $DANGER$ se le determinan sus k vecinos más cercanos en P y se opera de manera similar a SMOTE.
- "Borderline-SMOTE2" Han et al. (2005): es algoritmo es muy similar al descrito anteriormente, solo que éste no solo genera ejemplos a partir de los que están en $DANGER$ y sus vecinos positivos más cercanos en P, sino que también genera a partir de sus vecinos más cercanos de la clase mayoritaria en N (clase mayoritaria).
- "Safe-Level-SMOTE"Bunkhumpornpat et al. (2009): asigna a cada instancia minoritaria un nivel "seguro", antes de generar ejemplos sintéticos. Cada ejemplo sintético es situado cerca del mayor nivel "seguro", así de esta forma, todos los ejemplos sintéticos son generados solo en zonas seguras.
- SPIDER2 Napierala et al. (2010): Este algoritmo consta de dos fases, para preprocesar la clases mayoritaria y minoritaria respectivamente. En la primera fase, son identificadas las características de los ejemplos de la clase mayoritaria, y los ejemplos que sean ruidosos son eliminados o reetiquetados de clase. En la segunda fase, se identifican las características de los ejemplos de la clase minoritaria, teniendo en cuenta los cambios introducidos en la primera fase. Luego los ejemplos ruidosos de la clase minoritaria son incrementados a través de su replicación.

1.3.2. Algoritmos al nivel de los algoritmos

Las soluciones al nivel de los algoritmos son aquellas que tratan de adaptar un algoritmo de clasificación específico para reforzar el aprendizaje sobre la clase positiva. Estas soluciones

Figura 1.4: Ejemplo del funcionamiento de SMOTE-Tomeks links



pueden ser definidas como propuestas internas que crean nuevos algoritmos o modifican los existentes para tomar en consideración del desbalance de clases. Dada su relación con nuestro estudio futuro a continuación se describen dos ejemplos.

En Barandela et al. (2003) se propone el uso de una función de distancia ponderada para tener en cuenta el desbalance de clases a la hora de clasificar, esa función de distancia se describe:

Definición 3 *Función de distancia ponderada. Sea $d_E(\cdot)$ la métrica de Euclidean y Y es la nueva muestra a ser clasificada, x_0 es un ejemplo del conjunto de entrenamiento de la clase i, n_i es el número de ejemplos de la clase i, n es el número total de ejemplos en el conjunto de entrenamiento y m es la cantidad de características del conjunto. La función de distancia ponderada se define como:*

$$d_w(Y,x_0) = (n_i/n)^{1/m} d_E(Y,x_0) \tag{1.1}$$

La idea de este algoritmo es compensar el desbalance de clases sin tener que transformar el conjunto de entrenamiento (como en las técnicas al nivel de los datos). Los pesos son asignados a las clases y no a las instancias, a diferencia del algoritmo k-NN rule. De esta forma, dado que el factor de ponderación es mayor para la clase mayoritaria que para la clase minoritaria, la distancia a los ejemplos positivos se reduce mucho más que a los negativos, esto provoca que los nuevos ejemplos a clasificar encuentren más vecinos en la clase minoritaria y esta forma se mejora la clasificación de esta clase.

El algoritmo NWKNN (neighbor-weighted K-nearest neighbor) introducido en Tan (2005) para la clasificación de textos, sigue una filosofía similar al algoritmo descrito anteriormente. En el corpus tratado existen muchos más ejemplos de una clase que del resto, por lo tanto a la hora de clasificar, el algoritmo k-NN asigna la mayoría de los ejemplos hacia la clase más representada. Con el objetivo de mitigar ese problema se ponderan los ejemplos de la clase mayoritaria con valores pequeños y los ejemplos de la clase minoritaria con valores más altos. Este algoritmo funciona de la siguiente forma:

Para cada documento de prueba d, se seleccionan primeramente los K vecinos dentro de los documentos de entrenamiento contenidos en las $K*$ categorías $\left\{C_1^d, C_2^d, \ldots, C_{K*}^d\right\}$. El valor de peso es obtenido a través de la expresión:

$$Weigh_i = \frac{1}{\left(Num\left(C_i^d\right)/Min\left\{Num\left(C_l^d\right) \mid l = 1, \ldots, K*\right\}\right)^{1/Exponent}}$$

donde $Exponent > 1$. La regla de decisión ahora es mejorada a:

$$score\left(d, c_i\right) = Weight_i \left(\sum_{d_j \in KNN(d)} Sim\left(d, d_j\right) \delta\left(d_j, c_i\right) \right)$$

Por tanto la clasificación para un documento $d_j$con respcto a la clase c_i quedaría:

$$\delta\left(d_j, c_i\right) = \begin{cases} 1 & d_j \in c_i \\ 0 & d_j \notin c_i \end{cases}$$

De esta forma, tal como un k-NN tradicional, un nuevo documento d se asigna a la clase a la que la suma ponderada resulte mayor.

Como hemos visto en los algoritmo descritos anteriormente una variante que ha dado muy buenos resultados en contextos no balanceados es la ponderación de instancias y de clases atendiendo a su representatividad, favoreciendo a la clase minoritaria para de esta forma evitar el sesgo de los clasificadores tradicionales por la clase más representada.

1.3.3. Aprendizaje sensible al coste

Las soluciones del aprendizaje sensible al coste contemplan los cambios de coste a los diferentes tipos de clasificaciones incorrectas. Una matriz de coste codifica la penalización de clasificar ejemplos de una clase como que pertenecen a otra.$C(i, j)$ denota el coste de

predecir una instancia de la clase i como si fuera de j. Con esta notación, $C(+,-)$ es el coste de clasificar incorrectamente una instancia positiva (clase minoritaria) como negativa (clase mayoritaria) y $C(-,+)$ es el coste del caso contrario.

En dominios no balanceados la apreciación de las instancias positivas es mucho mayor que la de las instancias negativas, es por ello que el coste de clasificar mal un ejemplo positivo es siempre mayor que el de clasificar mal una negativa ($C(+,-) > C(-,+)$).

Cost-sensitive C4.5 decision tree Ting (2002): Es un algoritmo para construir árboles de decisión con sensibilidad al coste por clases, que tratan de minimizar el número de errores de alto coste y, a consecuencia de esto, permiten que se minimicen el total de ejemplos incorrectamente clasificados. El algoritmo cambia la distribución por clases, de forma tal que el árbol inducido está a favor de la clase con mayor ponderación/coste y así se logra que se cometan menos errores con alto coste.

Cost-sensitive SVM Veropoulos et al. (1999): Es un algoritmo es una modificación de la máquina de soporte vectorial "soft-margin". Se favorece el SVM de forma tal que se sitúa la frontera lejos de las instancias positivas, usando para ello diferentes costes para los errores de la clase positiva (C^+) y la negativa (C^-).

1.3.4. Algoritmos multiclasificadores

Los algoritmos multiclasificadores han mostrado un excelente comportamiento en dominios no balanceados Galar et al. (2012). En la mayoría de los casos una solución ensemble para datos no balanceados consiste en la combinación de variantes menores del mismo clasificador, los cuales son construidos a partir de la modificación de la distribución de los datos. En Galar et al. (2013) se introduce un algoritmo llamado EUSBOOST para conjuntos de alto desbalance. EUSBOOST es un algoritmo que usa bajo-muestreo evolutivo García and Herrera (2009) (Evolutionary UnderSampling EUS) guiado por boosting.

Esta propuesta es muy similar a RUSBOOST introducida en Seiffert et al. (2010), EUSBOOST utiliza bajo muestreo evolutivo. EUS surge de la aplicación de los algoritmos evolutivos para selección de prototipos en dominios no balanceados. En EUS cada

cromosoma es un vector binario que representa la presencia o no de una instancia en el conjunto de datos, la función de fitness que se utiliza tiene en cuenta la representatividad de ambas clases, es decir, el desbalance.

El comportamiento del algoritmo es estimado a través de una técnica "hold-one-out" y usando el clasificador 1NN, la métrica que se utiliza es la media geométrica (Geometric Mean GM), la cual posibilita maximizar la precisión en ambas clases a la misma vez, como se observa en la ecuación 1.2.

$$GM = \sqrt{TP_{rate} \cdot TN_{rate}} \tag{1.2}$$

La función de fitness del EUS originalmente propuesto se describe a continuación:

$$fitness_{EUS} = \begin{cases} GM - \left|\frac{p}{n} \cdot \rho\right| & if\ n > 0 \\ GM - \rho & if\ n = 0, \end{cases} \tag{1.3}$$

donde ρ es el factor de penalización que cuantifica la importancia dada al balance entre ambas clases.

Para el algoritmo EUSBOOST los autores proponen una nueva función fitness, modificando el procedimiento original (ecuación 1.3). La ecuación 1.4 muestra dicha función.

$$fitness_{EUS_Q} = fitness_{EUS} \cdot \frac{1{,}0}{\beta} \cdot \frac{10{,}0}{IR} - Q \cdot \beta \tag{1.4}$$

Donde $fitness_{EUS}$ es la función de fitness original (Equation 1.3), IR es el índice de desbalance, Q es el $Q - statistic$ global y β es el factor de ponderación cambiante en cada iteración:

$$\beta = \frac{N - t - 1}{N} \tag{1.5}$$

La diversidad global Q se decribe en la ecuación 1.6:

$$Q = \max_{i=1,\ldots,t} Q_{ij} \tag{1.6}$$

EUSBOOST incorpora dos nuevos pasos al algoritmo RUSBOOST y modifica un tercero, los pasos se explican en el algoritmo 1.1.

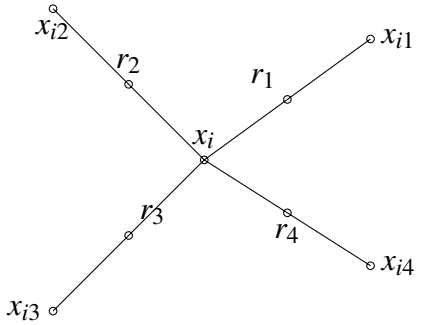
Algoritmo 1.1 EUSBoost, EUS ha sido incluido en AdaBoost.M2.

Entrada: Training set $S = \{x_i, y_i\}$, $i = 1, \ldots, N$; *and* $y_i \in \{c_1, c_2\}$
T: Number of iterations
I: Weak learner

Salida: Boosted classifier: $H(x) = \arg\max_{y \in C} \sum_{t=1}^{T} \ln\left(\frac{1}{\beta_t}\right) h_t(x,y)$ where $h_t \beta_t$ (with $h_t(x,y) \in [0,1]$)are the classifiers and their assigned weights,respectively

$D_1(i) \leftarrow \frac{1}{N}\, for\, 1 \ldots, N$
$w^1_{i,y} \leftarrow D_i(i)\ for\, i = 1, \ldots, N,\ y \neq y_i$
for $i \leftarrow 1$ **to** T **do**
  $W^1_{i,y} \leftarrow \sum_{y \neq y_i} w^t_{i,y}$
  $q_t(i,y) \leftarrow \frac{w^t_{i,y}}{W^t_i}\, for\, y \neq y_i$
  $D_t(i) \leftarrow \frac{\frac{W^t_i}{N}}{\sum_{i=i} W^t_i}$
  $S' =$ EvolutionaryUndersampling(S).
  $D'_t(k) \leftarrow \begin{cases} \frac{W^t_i}{\sum_{x_i \in S'} W^t_i} & if\, x_i \in S' \\ 0 & otherwise \end{cases}$
  $h_t \leftarrow I(S, D'_t)$
  $\varepsilon_t \leftarrow \frac{1}{2} \sum_{i=1} D_t(i) \left(1 - h_t(x_i, y_i) + \sum_{i, y \neq y_i}^{N} q_t(i,y)\, h_t(x_i, y)\right)$
  $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$
  $w^{t+1}_{i,y} = w^t_{i,y} \cdot \beta_t^{\left(\frac{1}{2}\right)(1 + h_t(x_i,y_i) - h_t(x_i,y))}\, for\, i = 1, \ldots, N,\ y \neq y_i$
end for

Figura 1.5: Una muestra de cómo SMOTE crea ejemplos sintéticos



EUSBOOST ha mostrado muy buen comportamiento en el ámbito de los no balanceados, mejorando considerablemente el resto de los multiclasificadores del estado del arte para no balanceados.

1.3.5. SMOTE: un algoritmo de generación de ejemplos sintéticos

Synthetic Minority Oversampling Technique (SMOTE) es un algoritmo que fue propuesto por Chawla y colaboradores en el 2002 Chawla et al. (2002), este algoritmo genera ejemplos sintéticos de la clase minoritaria, tomando cada ejemplo minoritario e introduciendo ejemplos sintéticos a lo largo del segmento que lo une con sus k vecinos más cercanos. Con esta propuesta la clase positiva es remuestreada a partir de cada uno de sus ejemplos interpolados con sus k vecinos más cercanos, dependiendo de la cantidad de ejemplos que se requiera generar, se escogen algunos vecinos aleatoriamente dentro de los k vecinos encontrados. Este proceso se muestra en la figura 1.5, donde $x_i$es el punto seleccionado, de x_{i1} a x_{i4}son algunos de los vecinos más cercanos seleccionados, y de r_1 a r_4 son los de datos sintéticos creados a través de la interpolación aleatoria.

Se toma la diferencia entre el vector de atributos (muestra) del ejemplo seleccionado y sus vecinos más cercanos. Se multiplica esta diferencia por un número aleatorio entre 0 y 1, esto se le suma al vector de atributos del ejemplo seleccionado. Con esto se logra la selección de un punto aleatorio en la línea entre el ejemplo sus vecinos. El algoritmo está detallado en la figura 1.2, y consta de los siguientes pasos:

1. Paso 1: Seleccionar el número de ejemplos de la clase minoritaria que serán usados para generar nuevas instancias cuando el grado de sobremuestreo es menor que el 100 %.
2. Paso 2: determina todos los k-vecinos de las instancias que serán replicadas.
3. Paso 3: Finalmente se realiza la interpolación como se explicó anteriormente. Para ganar en claridad se muestra la figura 1.6.

Algoritmo 1.2 Algoritmo SMOTE

Entrada: Number of minority class samples T
Number of attributes *nattrs*
Array for the original minority class samples $Sample[][]$
Amount of SMOTE $N\%$
Number of nearest neighbors k
Salida: $Synthetic[]$ an array of $(N/100)*T$ synthetic minority class samples

if $N < 100$ **then**
  Ramdomize the T minority samples
  Fill $Sample[][]$ wih the fiTCA $(N/100)*T$ samples
  $T = (N/100)*T$
  $N = 100$
end if
for $i \leftarrow 1$ **to** T **do**
  Compute k neaTCA neighbors for i, and save the indeces in an array *nnarray*
  $Populate(N, i, nnarray)$
end for
while $N \neq 100$ **do**
  Choose a ramdom number between 1 and k, call it *nn*
  for $attr \leftarrow 1$ **to** *nattrs* **do**
    $Compute : dif = Sample[nnarray[nn]][attr]] - Sample[i][attr]$
    $Compute : gap = random\ number\ between\ 0\ and\ 1$
    $Synthetic[newindex][attr] = Sample[i][attr] + gap * dif$
  end for
  $newindex++$
  $N = N - 1$
end while

A diferencia de las técnicas comunes para replicar ejemplos (ej. el sobremuestreo aleatorio), en las cuales la región de decisión llega a ser usualmente más específica, con

Figura 1.6: Ejemplo de la aplicación de SMOTE

```
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: f11 = 6 f21 = 4, f21 - f11 = -2
f12 = 4 f22 =3,  f22 - f12 = -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.
```

SMOTE el problema del sobreaprendizaje se evita al extender la frontera de la clase minoritaria hacia la región de la clase mayoritaria, a través de la creación de ejemplos de la clase minoritaria a partir de los cuales aprender.

Finalmente enfatizamos que la implementación de SMOTE que se utiliza en esta tesis es la que aparece implementada en la herramienta de código abierto KEEL [1]Alcalá et al. (2009, 2010).

1.3.6. Métricas de evaluación en dominios no balanceados

El comportamiento de los algoritmos de aprendizaje automático es usualmente evaluado usando precisión predictiva. Sin embargo esto no es apropiado cuando el conjunto no está balanceado y/o cuando el costo de los diferentes errores varía marcadamente Chawla et al. (2004b).

Weiss y Hirsh en Weiss and Hirsh (2000) mostraron que el porciento de error de la clasificación de las reglas de la clase minoritaria resulta dos o tres veces menor que el de las reglas que identifican los ejemplos de la clase mayoritaria, y que existe menos probabilidad de predecir ejemplos positivos que negativos. Es por ello que en lugar de usar porciento de error o porciento de instancias correctamente clasificadas, en el contexto de los no balanceados son consideradas otras métricas más apropiadas.

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

Una matriz de confusión es una forma de tabla de contingencia que muestra las diferencias entre la clase verdadera y la clase predicha para un conjunto de ejemplos etiquetados. Dicha matriz induce las métricas bien conocidas: verdadero positivo (true positive TP), verdadero negativo (true negative TN), falso positivo (false positive) y falso negativo (false negative).

En la tabla 1.1 se muestra un ejemplo de matriz de confusión.

Tabla 1.1: Matriz de confusión para un problema de 2 clases

	Predicción positiva	Predicción negativa
Clase positiva	true positive (TP)	False negative (FN)
Clase negativa	False positive (FP)	True negative (TN)

A partir de esta tabla se obtienen las métricas clásicas de error y precisión de la siguiente forma:

$$Error\,rate = \frac{FP+FN}{TP+FP+TN+FN} \tag{1.7}$$

$$Precision = \frac{TP+TN}{TP+FP+TN+FN} = 1 - Error\,rate \tag{1.8}$$

También es posible derivar cuatro métricas de comportamiento que evalúan directamente el comportamiento de la clasificación de las clases positiva y negativa por independiente:

- **True postive rate** $TP_{rate} = \frac{TP}{TP+FN}$ constituye el porciento de casos positivos correctamente clasificados que pertenecen a la clase postiva.
- **True negative rate** $TN_{rate} = \frac{TN}{FP+TN}$ constituye el porciento de casos negativos correctamente clasificados que pertenecen a la clase negative.
- **False positive rate** $FP_{rate} = \frac{FP}{FP+TN}$ constituye el porciento de ejemplos negativos que fueron mal clasificados como ejemplos positivos.
- **False negative rate** $FN_{rate} = \frac{FN}{TP+FN}$ constituye el porciento de ejemplos positivos que fueron mal clasificados como ejemplos negativos.

Estas cuatro métricas de comportamiento tienen la ventaja de ser independientes del costo por clases y las probabilidades a priori. El principal objetivo de un clasificador es minimizar el número de falsos positivos y negativos, y de igual forma, maximizar el número de verdaderos positivos y negativos.

Una métrica apropiada que pudiera ser usada para medir el comportamiento de la clasificación sobre conjuntos no balanceados es la gráfica del Receiver Operating Characteristic (ROC) Bradley (1997). En esta gráfica, se visualiza la trayectoria entre el beneficio (TP_{rate}) y el coste (FP_{rate}), y esto demuestra el hecho de que ningún clasificador puede incrementar el número de verdaderos positivos sin aumentar también el número de falsos positivos. El área bajo la curva ROC (AUC) Huang and Ling (2005) corresponde a la probabilidad de identificar correctamente entre dos estímulos cuál es ruido y cuál tiene la señal más ruido. AUC resume en un número el comportamiento de los algoritmos de aprendizaje.

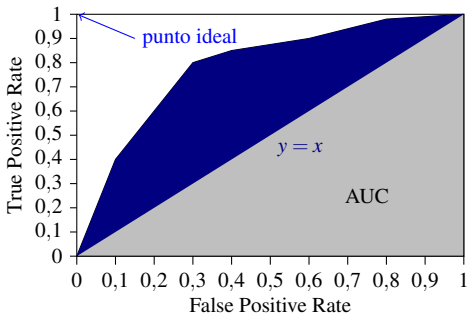
La forma de construir la curva ROC es graficar en 2 dimensiones el TP_{rate} (eje Y) contra el FP_{rate} (eje X) como se muestra en la figura 1.7. Los puntos $(0,0)$ y $(1,1)$ se corresponde con un clasificador trivial, en el cual la clase es siempre predicha como negativa y positiva respectivamente, mientras que el punto $(0,1)$ representa una clasificación perfecta. Para calcular el AUC, solo es necesario obtener el área bajo la gráfica que se muestra en la figura 1.7, a través de la expresión 1.9

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \tag{1.9}$$

1.4. Teoría de los Conjuntos Aproximados

La Teoría de Conjuntos Aproximados, conocida en inglés como Rough Set Theory, fue introducida por Zdzislaw Pawlak en un artículo publicado en 1982 Pawlak (1982). Actualmente la TCA provee una metodología para tratar diferentes tipos de problemas como vaguedad, indecisión, imprecisión, entre otras; todas causadas por inconsistencias en los datos Bello et al. (2008).

Figura 1.7: Área bajo la curva ROC



La TCA es una excelente herramienta matemática para modelar la incertidumbre cuando esta se manifiesta en forma de inconsistencia, permite tratar tanto datos cuantitativos como cualitativos, y no se requiere eliminar las inconsistencias previas al análisis; respecto a la información de salida pues puede ser usada para determinar la relevancia de los atributos, generar las relaciones entre ellos (en forma de reglas), entre otras, Choubey (1996); Chouchoulas and Shen (1999); Grzymala-Busse (1994); Miao and Hou (2003); Skowron (1999); Tsumoto (2003).

Esta teoría se basa en aproximar cualquier concepto, un subconjunto del dominio como por ejemplo, una clase en un problema de clasificación supervisada, por un par de conjuntos exactos, llamados aproximación inferior y aproximación superior del concepto.

En siguientes subsecciones se ofrecen las principales definiciones de la TCA, además se detalla lo relacionado con la TCA basada en relaciones de similaridad.

Definiciones de la Teoría de los Conjuntos Aproximados

La filosofía de la TCA se basa en la suposición de que con todo objeto x de un universo U está asociada una cierta cantidad de información, expresada por medio de atributos que describen el objeto. En la TCA la estructura de información básica es el Sistema de Información.

Definición 4 *Sistema de Información y sistema de decisión. Sea un conjunto de atributos* $A = \{a_1, a_2, ...a_n\}$ *y un conjunto* U *no vacío llamado universo de ejemplos (objetos,*

entidades, situaciones o estados) descritos usando los atributos a_i; al par (U,A) se le denomina sistema de información. Si a cada elemento de U se le agrega un nuevo atributo d llamado decisión, indicando la decisión tomada en ese estado o situación, entonces se obtiene un Sistema de decisión$(U,A\cup\{d\}\,d)$, donde $d\in A$.

Definición 5 *Función de información. A cada atributo a_i se le asocia un dominio v_i. Se tiene una función $f:UxA\leftarrow V$, $V=\{v_1,v_2,...v_p\}$ tal que $f(x,a_i)\in v_j$ para cada $a_i\in A$, $x\in U$, llamada función de información.*

El atributo de decisión d induce una partición del universo U de objetos. Sea el conjunto de enteros $\{1,...,l\}$, $X_i=\{x\in U:\ d(x)=i\}$, entonces $\{X_1,...,X_l\}$ es una colección de clases de equivalencias, llamadas clases de decisión, donde dos objetos pertenecen a la misma clase si ellos tienen el mismo valor para el atributo decisión.

Se dice que un atributo $ai\in A$ separa o distingue un objeto x de otro y, y se escribe $Separa(a_i,x,y)$, si y solo si se cumple:

$$f(x,a_i)\neq f(y,a_i) \tag{1.10}$$

La relación de separabilidad se basa en la comparación de los valores de un atributo, para lo cual se ha usado la igualdad (o desigualdad) estricta. Sin embargo, es posible usar una condición de comparación menos estricta definida de esta forma:

$$Separa(a_i,x,y)\Longleftrightarrow|f(x,a_i)-f(y,a_i)|>\varepsilon \tag{1.11}$$

Definición 6 *Relación de inseparabilidad. A cada subconjunto de atributos B de A $B\subseteq A$ está asociada una relación binaria de inseparabilidad denotada por R, la cual es el conjunto de pares de objetos que son inseparables uno de otros por esa relación*

$$R=\{(x,y)\in UxU\ :\ f(x,a_i)=f(y,a_i)\,\forall a_i\in B\} \tag{1.12}$$

Una relación de inseparabilidad (indiscernibility relation) que sea definida a partir de formar subconjuntos de elementos de U que tienen igual valor para un subconjunto de atributos B de A, $B \subseteq A$, es una relación de equivalencia.

Los principales conceptos de la teoría son las aproximaciones inferior y superior de un subconjunto $X \subseteq U$. Estos conceptos fueron originalmente introducidos con referencia a una relación de inseparabilidad R. Sea R una relación binaria definida sobre U la cual representa la inseparabilidad, se dice que $R(x)$ significa el conjunto de objetos los cuales son inseparables de x. Así, $R(x) = \{y \in U : yRx\}$. En la TCA clásica, R es definida como una relación de equivalencia; es decir, es una relación binaria $R \subseteq UxU$ que es reflexiva, simétrica y transitiva. R induce una partición de U en clases de equivalencia correspondiente a $R(x)$, $x \in U$.

Este enfoque clásico de TCA es extendido mediante la aceptación que objetos que no son inseparables pero sí suficientemente cercanos o similares puedan ser agrupados en la misma clase Slowinski and Vanderpooten (1997). El objetivo es construir una relación R' a partir de la relación de inseparabilidad R pero flexibilizando las condiciones originales para la inseparabilidad. Esta flexibilización puede ser realizada de múltiples formas, así como pueden ser dadas varias definiciones posibles de similitud. Existen varias funciones de comparación de atributos (funciones de similitud), las cuales están asociadas al tipo del atributo que se compara Wilson and Martinez (1997). Sin embargo, la relación R' debe satisfacer algunos requerimientos mínimos. Si R es una relación de inseparabilidad definida en U, R' es una relación de similitud extendida de R si y solo si $\forall x \in U$, $R(x) \subseteq R'(x)$y $\forall x \in U$, $\forall y \in R'(x)$, donde $R'(x)$ es la clase de similitud de x, es decir, $R'(x) = \{y \in U : yR'(x)\}$. R' es reflexiva, cualquier clase de similitud puede ser vista como un agrupamiento de clases de inseparabilidad y R' induce un cubrimiento de U Skowron and Stepaniuk (1992). Esto muestra que un objeto puede pertenecer a diferentes clases de similitud simultáneamente, lo que significa que el cubrimiento inducido por R' sobre U no es necesariamente una partición.

La aproximación de un conjunto $X \subseteq U$, usando una relación de inseparabilidad R, ha sido inducida como un par de conjuntos llamados aproximaciones $R-inferior$ y $R-superior$ de X. Se considera en esta investigación una definición de aproximaciones más general,

la cual maneja cualquier relación reflexiva R'. Las aproximaciones $R' - inferior\ (R'_*(X))$ y $R' - superior(R'^*(X))$ de X están definidas respectivamente como se muestra en las expresiones 1.13 y 1.14.

$$R'_*(X) = \{x \in X\} : \left|R'(x) \subseteq X \right. \tag{1.13}$$

$$R'^*(X) = \bigcup_{x \in X} R'(x) \tag{1.14}$$

Teniendo en cuenta las expresiones definidas en 1.13 y 1.14, se define la región límite de X para la relación R' Deogun (1995):

$$BN_B(X) = R'^*(X) - R'_*(X) \tag{1.15}$$

Si el conjunto BN_B es vacío entonces el conjunto X es exacto respecto a la relación R'. En caso contrario, $BN_B(X) \neq \emptyset$, el conjunto X es inexacto o aproximado con respecto a R'. El uso de relaciones de similitud ofrece mayores posibilidades para la construcción de las aproximaciones; sin embargo, se tiene que pagar por esta mayor flexibilidad, pues es más difícil desde el punto de vista computacional buscar las aproximaciones relevantes en este espacio mayor. Existen varias funciones de comparación de atributos (funciones de similitud), las cuales están asociadas al tipo de los atributos. Algunas de estas funciones de similitud se describen en Wilson and Martinez (1997). En el siguiente epígrafe se abordará más sobre este tema.

A partir de las aproximaciones inferior y superior de un concepto X se definen tres regiones para caracterizar el espacio de aproximación: la región positiva que es la aproximación $R' - inferior$, la región límite que es el conjunto BN_B y la región negativa $NEG(X)$ que es la diferencia entre el universo y la aproximación $R' - superior$. Los conjuntos $R'_*(X)$ (denotado también como $POS(X)$), $R'^*(X)$, $BN_B(X)$ y $NEG(X)$ son las nociones principales de la Teoría de Conjuntos Aproximados.

CAPÍTULO 2

Un algoritmo de preprocesamiento basado en SMOTE y la Teoría de los Conjuntos Aproximados para la limpieza de ejemplos artificiales para no balanceo muy alto

"It's not because things are difficult that we dare not venture. It' s because we dare venture that they are difficult"

– Lucio Anneo Séneca.

En este capítulo se introduce un nuevo algoritmo de preprocesamiento híbrido para conjuntos de datos de alto desbalance que combina el algoritmo SMOTE y un algoritmo de limpieza basado en la TCA.

Para la creación de nuestra propuesta hemos partido de las siguientes dos premisas:

1. Los conjuntos de alto desbalance tienen especial relevancia, es decir cuando el índice de desbalance (definido como la fracción entre el número de ejemplos de la clase

mayoritaria entre los de la minoritaria) es alto (mayor o igual que 9, pues la clase minoritaria representa menos del 10 %)

2. El algoritmo SMOTE puede introducir ejemplos sintéticos de la clase minoritaria en el área de la clase mayoritaria. Este fenómeno se conoce como "sobregeneralización". Para intentar resolver este problema, se han aplicado algunos algoritmos de limpieza, tales como ENN y TomekLinks Batista et al. (2004), Borderline Han et al. (2005) y Safelevel Bunkhumpornpat et al. (2009).

La idea básica de nuestro algoritmo es que los nuevos ejemplos creados, antes de pasar a formar parte del conjunto de entrenamiento final, sean evaluados teniendo en cuenta su pertenencia o no a la aproximación inferior de su clase, y de esta forma garantizar una mejor separabilidad de clases. El algoritmo solo evalúa los ejemplos sintéticos creados, los ejemplos originales permanecen intactos.

En este capítulo se ofrecen los elementos teóricos relacionados con la TCA basada en relaciones de similaridad, se describe en detalles nuestra propuesta, así como la configuración del algoritmo, el ajuste de parámetros, los conjuntos de datos utilizados, las pruebas estadísticas y el estudio experimental realizado, el cual conduce a las conclusiones que se realizan al final del mismo.

2.1. SMOTE-RSB*: un algoritmo de preprocesamiento que combina SMOTE y Teorías de los Conjuntos Aproximados

En este epígrafe se presenta una nueva propuesta para lograr que la distribución de ejemplos en las clases en un conjunto de datos no balanceado sea uniforme.

Como se estudió en el capítulo 1, a pesar de que existen varios algoritmos que funcionan sobre la base de generar ejemplos sintéticos usando SMOTE y luego aplicar un algoritmo de limpieza para evitar la "sobregeneralización", éstos no logran garantizar que los ejemplos que resulten seleccionados sean verdaderamente representativos, debido a que su selección está basada generalmente en criterios de vecindad.

En la sección 1.4 estudiamos la TCA y su aplicación en la edición de datos, debido a su capacidad para eliminar inconsistencias. Esta potente herramienta matemática, a través de las 3 particiones que crea en el universo, es capaz de decantar los ejemplos en: los que no son buenos representantes de su clase (región negativa), los que son dudosos (frontera) y los que se pudieran considerar como buenos representantes de su clase (región positiva). En la subsección 2.1.1 estudiaremos la extensión de la TCA basada en relaciones de similaridad, dichos conceptos son aplicados en la determinación de la aproximación inferior que utiliza nuestra propuesta.

Nuestra hipótesis es que algunas de las instancias sintéticas introducidas pueden no ser apropiadas para ser usadas en la fase de aprendizaje, por lo que eliminamos todas aquellas que no pertenezcan a la aproximación inferior de su clase. Nuestro procedimiento se les aplica a las instancias sintéticas, mientras que las originales permanecen intactas.

2.1.1. Teoría de los Conjuntos Aproximados basados en relaciones de similaridad

La TCA basada en relaciones de similaridad considera que 2 objetos son inseparables si éstos son similares con un valor mayor o igual que un umbral dado.

Este valor de umbral requiere especial atención dado que este valor es el usado para determinar la similaridad entre objetos para obtener la aproximación inferior y la superior.

La expresión usada para determinar el grado de similaridad entre 2 objetos x_i e y_i se define de la siguiente forma (guardando cada valor en una matriz de similaridad $SimilarityMatrix$):

$$SimilarityMatrix(i,j) = \frac{\sum_{k=1}^{n} w_k * \delta_k(x_{ik}, x_{jk})}{M} \tag{2.1}$$

donde n es el número de atributos, w_k es el valor de peso asignado a cada atributo k, x_{ik} y x_{jk} son los valores de los atributos k respectivamente, δ_k es la función de comparación de atributos para k, M es el número de atributos considerados en el relación de equivalencia, B es el conjunto de atributos en la relación de equivalencia.

Donde el valor de peso de cada atributo se define como:

$$w_k = \begin{cases} 1 & if\ k \in B \\ 0 & other\ case \end{cases} \tag{2.2}$$

δ_k para atributos discretos se calcula de la siguiente forma:

$$\delta_k(x_{ik}, x_{jk}) = \begin{cases} 1 & if\ x_{ik} = x_{jk} \\ 0 & other\ case \end{cases} \tag{2.3}$$

y para atributos continuos:

$$\delta_k(x_{ik}, x_{jk}) = 1 - \frac{\left| x_{ik} - x_{jk} \right|}{maxA_k - minA_k} \tag{2.4}$$

donde $maxA_k$ y $minA_k$ son los valores extremos (intervalo) del atributo k.

2.1.2. Algoritmo SMOTE-RSB*

El algoritmo que se presenta en el presente capítulo hace uso de la región positiva para decidir cuáles de los ejemplos artificiales creados por SMOTE pasarán al conjunto final.

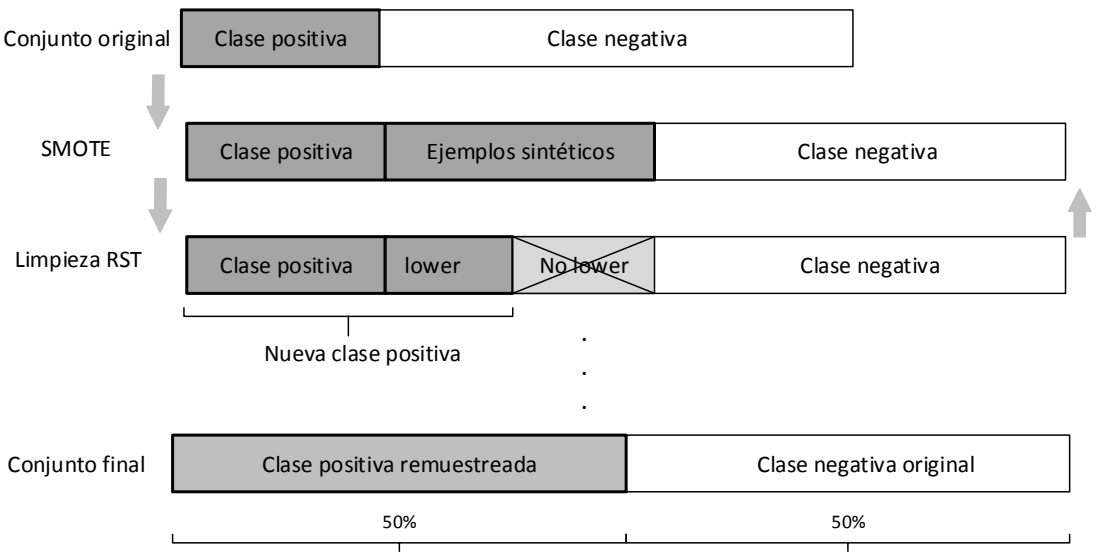
Este algoritmo híbrido consta de 2 estados principales:

1. Primero, se crean instancias sintéticas de la clase minoritaria usando para ello el algoritmo SMOTE explicado en detalles en el epígrafe 1.3.5.
2. Segundo, se le aplica al conjunto resultante del paso 1 un algoritmo de limpieza basado en la TCA, que inserta en el conjunto final (llamado *resultset*) solo aquellos ejemplos sintéticos que pertenecen a la aproximación inferior de su clase (B_*).

La figura 2.1 muestra un esquema del funcionamiento el algoritmo $SMOTE - RSB^*$.

El algoritmo de limpieza usado está basado en una extensión de la TCA basado en relaciones de similaridad, el mismo tiene cinco pasos que se detallan a continuación:

Figura 2.1: Esquema del funcionamiento de SMOTE-RSB*



1. "Paso 1": usar SMOTE para generar ejemplos sintéticos de la clase minoritaria hasta que ambas clases queden balanceadas (este paso se corresponde con el estado 1).
2. "Paso 2": construir el conjunto resultante (salida del algoritmo) incluyendo solo los ejemplos originales.
3. "Paso 3": construir la matriz de similaridad para todas instancias en el conjunto , usando para ello la expresión 2.1.
4. "Paso 4": analizar cuáles de los nuevos ejemplos sintéticos pertenece a la aproximación inferior. Insertar estos objetos en el "resultSet".
5. "Paso 5": Si no se encuentran ejemplos en el paso 4, entonces el conjunto resultante es como lo devuelve SMOTE.

La figura 2.1 muestra el algoritmo asociado a los pasos descritos previamente.

Los valores de umbral fueron definidos durante la experimentación.

Algoritmo 2.1 Algoritmo propuesto SMOTE-RSB*

Entrada: Initial threshold It
Final threshold Ft
SMOTE parameters
Salida: $resultSet$

1: Step1: Using SMOTE we create a set of synthetic data ($syntheticInstances[]$) for the minority class until the training set is balanced.
2: Step2: Create $resultSet$, including the original instances.
3: Step3: Construct the $similarityMatrix$ for the synthetic instances organized in rows and negative instances (majority class) in columns, considering all the features in the equivalence relation. In the $similarityMatrix(i, j)$ we will find the degree between instances i and j.
4: Step4: For every synthetic example count the number of similar examples in the negative class.
5: $npos - syn$: number of synthetic instances
6: $similarityValue = It$
7: **while** ($resultSet\ is\ empty$) & ($Similarityvalue <= Ft$) **do**
8: **for** $i \leftarrow 1$ **to** $npos - syn$ **do**
9: **for** $j \leftarrow 1$ **to** $nneg$ **do**
10: **if** $similarityMatrix(i, j)$>$similarityValue$ **then**
11: $cont[i]++$
12: **end if**
13: **end for**
14: **if** $cont[i]$=0 **then**
15: insert $syntheticInstances[i]$ in $resultSet$
16: **end if**
17: **end for**
18: $similarityValue += 0{,}05$
19: **end while**
20: Step5: If there are no instances in the lower approximation, the solution is given as the set balanced with SMOTE, all synthetic instances are include in $resultSet$.

2.2. Configuración del estudio experimental: conjuntos, parámetros y pruebas estadísticas

En este epígrafe se describirá el estudio experimental llevado a cabo para demostrar la viabilidad de la propuesta. Primeramente serán descritos los conjuntos de datos que se usarán en los experimentos, así como los parámetros y las configuraciones de los algoritmos usados. Se describirán las pruebas estadísticas seleccionadas para encontrar diferencias entre los algoritmos. El algoritmo de aprendizaje que se ha seleccionado para el estudio es el árbol de decisión C4.5 Quinlan (1993), este algoritmo se encuentra dentro de los 10 más influyentes en minería de datos X.Wu et al. (2008) y además ha sido muy usado en problemas de aprendizaje a partir de datos no balanceados Batista et al. (2004).

2.2.1. Conjuntos de datos y parámetros

Para analizar la propuesta presentada, se han seleccionado 44 conjuntos de datos del repositorio de la UCI Asuncion and Newman (2007) que tienen un alto índice de desbalance (mayor que 9). Estos conjuntos fueron construidos a partir de conjuntos de múltiples clases, los cuales fueron modificados para llevarlos a 2 clases, la unión de una o más clases poco representadas conformaron la clase minoritaria y el resto de las clases fueron etiquetadas como mayoritaria. Estos conjuntos están descritos en la tabla 2.1, la columna IR indica el índice de desbalance obtenido a través de la expresión 2.5.

$$IR = \frac{\#instancias\ mayoritarias}{\#instancias\ minoritarias} \tag{2.5}$$

Cada conjunto ha sido particionado con el objetivo de realizar una validación cruzada usando 5 particiones (fivefolds cross-validation), el 80 % es usado como conjunto de entrenamiento y el 20 % como conjunto de prueba, donde las 5 particiones de prueba conforman el conjunto completo. Para cada conjunto el resultado considerado es el promedio de las 5 particiones. Las particiones fueron realizadas de forma tal que el número de instancias por clase permaneciera uniforme Batista et al. (2004). Los conjuntos de

Tabla 2.1: Descripción de los conjuntos de datos usados en los experimentos

Conjuntos	# Ej	#Atributos	%Class(min., maj.)	IR
ecoli0137vs26	281	7	(2.49, 97.51)	39.15
shuttle0vs4	1829	9	(6.72, 93.28)	13.87
yeastB1vs7	459	7	(6.53, 93.47)	14.3
shuttle2vs4	129	9	(4.65, 95.35)	20.5
glass016vs2	192	9	(8.85, 91.15)	10.29
glass016vs5	184	9	(4.89, 95.11)	19.44
pageblocks13vs4	472	10	(5.93, 94.07)	15.85
yeast05679vs4	528	8	(9.66, 90.34)	9.35
yeast1289vs7	947	8	(3.16, 96.84)	30.5
yeast1458vs7	693	8	(4.33, 95.67)	22.10
yeast2vs4	514	8	(9.92, 90.08)	9.08
Ecoli4	336	7	(6.74, 93.26)	13.84
Yeast4	1484	8	(3.43, 96.57)	28.41
Vowel0	988	13	(9.01, 90.99)	10.10
Yeast2vs8	482	8	(4.15, 95.85)	23.10
Glass4	214	9	(6.07, 93.93)	15.47
Glass5	214	9	(4.20. 95.80)	22.81
Glass2	214	9	(7.94, 92.06)	11.59
Yeast5	1484	8	(2.96, 97.04)	32.78
Yeast6	1484	8	(2.49, 97.51)	39.16
abalone19	4174	8	(0.77, 99.23)	128.87
abalone918	731	8	(5.65, 94.25)	16.68
cleveland0vs4	177	13	(7.34, 92.66)	12.61
ecoli01vs235	244	7	(2.86,97.14)	9.16
ecoli01vs5	240	7	(2.91,97.09)	11
ecoli0146vs5	280	7	(2.5,97.5)	13
ecoli0147vs2356	336	7	(2.08,97.92)	10.58
ecoli0147vs56	332	7	(2.1,97.9)	12.28
ecoli0234vs5	202	7	(3.46,96.54)	9.1
ecoli0267vs35	224	7	(3.12,96.88)	9.18
ecoli034vs5	300	7	(2.33,97.67)	9
ecoli0346vs5	205	7	(3.41,96.59)	9.25
ecoli0347vs56	257	7	(2.72,97.28)	9.28
ecoli046vs5	203	7	(3.44,96.56)	9.15
ecoli067vs35	222	7	(3.15,96.85)	9.09
ecoli067vs5	220	7	(3.18,96.82)	10
glass0146vs2	205	9	(4.39,95.61)	11.05
glass015vs2	172	9	(5.23,94.77)	9.11
glass04vs5	92	9	(9.78,90.22)	9.22
glass06vs5	108	9	(8.33,91.67)	11
led7digit02456789vs1	443	7	(1.58,98.42)	10.97
yeast0359vs78	506	8	(9.8, 90.2)	9.12
yeast0256vs3789	1004	8	(9.86, 90.14)	9.14
yeast02579vs368	1004	8	(9.86, 90.13)	9.14

datos particionados están disponibles en la página web de KEELAlcalá et al. (2009, 2010) (KEEL-dataset), en el hipervínculo: http://www.keel.es/datasets.php.

En los experimentos se usaron los siguientes parámetros para el algoritmo $SMOTE - RSB^*$:

- k: número de vecinos más cercanos, fijado en 5.
- Función de distancia para obtener los vecinos más cercanos, se usó la distancia de Euclidean
- Distribución por clases para el remuestreo, se definió 50 %-50 %.

Estos son los parámetros recomendados por los autores del algoritmo SMOTE Chawla et al. (2002), y por tanto se han usado como estándares en los experimentos.

Para los algoritmos del estado del arte los parámetros usados son los que aparecen por defecto en la herramienta KEEL, los mismos se muestran en la tabla 2.2:

2.2.2. Pruebas estadísticas

En esta investigación se utilizan técnicas de pruebas de hipótesis para dar soporte estadístico al análisis de resultados que se realiza García et al. (2009); Sheskin (2003). Específicamente se utilizan pruebas no paramétricas, debido al hecho de que las condiciones iniciales que garantizan la consistencia de una prueba paramétrica no son satisfechas, y esta forma la prueba estadística perdería credibilidad Demsâr (2006).

Para múltiples comparaciones, se usa una prueba de Iman-Davenport Iman and Davenport (1980), la cual encuentra diferencias estadísticas entre un grupo de resultados, y una prueba post hoc de HolmHolm (1979) con el objetivo de encontrar para qué algoritmos se rechaza la hipótesis de igualdad con respecto al algoritmo seleccionado como algoritmo de control.

El procedimiento post hoc permite conocer cuando una hipótesis de comparación de medias puede ser rechazado con un nivel específico de significación α. Sin embargo resulta muy interesante calcular el p-value asociado a cada comparación, el cual representa el menor

Tabla 2.2: Parámetros usados para los algoritmos del estado del arte

Algoritmos	Parámetros
SMOTE	Number of Neighbors = 5 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function = Euclidean Type of Interpolation = standard
SMOTE-TomekLinks	Number of Neighbors = 5 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function = Euclidean
SMOTE-ENN	Number of Neighbors ENN = 3 Number of Neighbors SMOTE = 5 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function (SMOTE) = Euclidean Distance Function (ENN) = Euclidean
Borderline-SMOTE1	Number of Neighbors SMOTE = 5 Number of Neighbors for considering a instance border = 3 Type of borderline SMOTE = 1 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function (SMOTE) = Euclidean Type of Interpolation = standard
Borderline-SMOTE2	Number of Neighbors SMOTE = 5 Number of Neighbors for considering a instance border = 3 Type of borderline SMOTE = 1 Type of SMOTE = both Balancing = YES Quantity of generated examples = 2 Distance Function (SMOTE) = Euclidean Type of Interpolation = standard
Safe-Level-SMOTE	Number of Neighbors = 5 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function = Euclidean Type of Interpolation = standard

nivel de significación de una hipótesis que resulta rechazada. De esta forma es posible saber cuándo 2 algoritmos son significativamente diferentes y cuán diferentes son.

De igual forma, se considera el promedio del ranking de los algoritmos que se comparan, con el objetivo de mostrar gráficamente cuán bueno es un algoritmo con respecto al resto. Este ranking es obtenido asignando una posición a cada algoritmo, dependiendo de su comportamiento, para cada conjunto de datos. El algoritmo que obtenga el mejor AUC, detallado en la sección 1.3.6, para un conjunto específico obtiene el primer ranking (valor 1); el algoritmo con el segundo mejor resultado se le asigna el ranking 2, y así sucesivamente. Este procedimiento se lleva a cabo con todos los conjuntos y finalmente se calcula un promedio del ranking como un valor medio de todos los rankings.

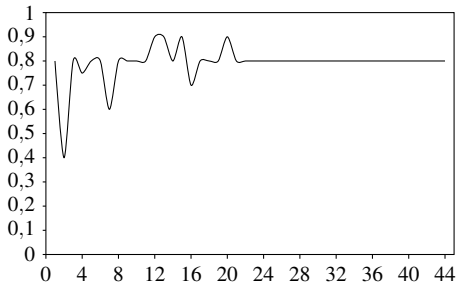
Estas pruebas han sido sugeridas en varios estudios presentados en Demsâr (2006); García and Herrera (2008); García et al. (2009, 2010), donde su uso es altamente recomendado en campo del aprendizaje automático. Cualquier lector interesado puede encontrar más información al respecto en sitio web http://sci2s.ugr.es/sicidm/, junto con el software que realiza dichas pruebas estadísticas.

2.3. Análisis de los parámetros de $SMOTE - RSB^*$

Como se explicó en la sección 2.1, el valor *similarityvalue* fue fijado durante la ejecución del algoritmo. El algoritmo comienza con valor inicial (It) en 0,4, el algoritmo de limpieza no encuentra no encuentra ninguna instancia en la aproximación inferior con ese valor, lo incrementa en 0,05 mientras sea menor que el umbral final (Ft) fijado en0,9. De esta se garantiza que en todos los conjuntos se busquen los objetos en la aproximación inferior con el menor valor posible, garantizando buena calidad de ejemplos sintéticos que se inserten en el conjunto final.

La figura 2.2 muestra el máximo valor de similaridad con el que cada conjunto queda balanceado, cada valor en el eje de las x representa un conjunto de datos, mientras que cada valor en el eje de las y representa un valor de similaridad. Como se puede observar la mayoría de los conjuntos queda balanceado cuando el valor de similaridad es cercano a 0,8, específicamente los de mayor índice de desbalance

Figura 2.2: Valor de similaridad para cada conjunto de datos



2.4. Análisis comparativo

En este epígrafe se comparará el algoritmo $SMOTE - RSB^*$ con otros 6 algoritmos de preprocesamiento reconocidos dentro estado del arte, todos basados en SMOTE, los cuales fueron explicados detalles en el epígrafe 1.3.1. Es decir, se realizará la comparación con el propio SMOTE y con 5 hibridaciones de éste con algoritmos de limpieza: SMOTE-TomekLinks, SMOTE-ENN, Borderline-SMOTE1, Borderline-SMOTE2 y Safe-Level-SMOTE.

En las tablas 2.3 y 2.4 se muestran los resultados del experimental en cuanto a training y test respectivamente. En la primera columna se muestran los resultados del conjunto original (sin preprocesamiento), el mejor algoritmo para cada conjunto de datos aparece resaltado en negrita.

Como se puede observar $SMOTE - RSB^*$ obtiene mejores resultados que el resto de los algoritmos. Por otra parte resaltan los buenos resultados alcanzados por los algoritmos SMOTE-TomekLinks y SMOTE-ENN, lo cual enfatiza en las ventajas de aplicar un algoritmo de limpieza después del sobremuestreo para lograr mejores resultados en la etapa de clasificación. Finalmente se observa que los algoritmos de preprocesamiento obtienen mejores resultados que el conjunto original.

La tabla 2.5 muestra el número de veces que cada algoritmo comparado obtiene el mejor AUC. Se muestran dos números por cada celda, el primero corresponde al número de veces que el algoritmo resulta ganador absoluto, mientras que el segundo denota cuántas

Tabla 2.3: Comparación del AUC alcanzado por la propuesta y 6 algoritmos de preprocesamiento en training

Dataset	Original	Smote	S-TL	S-ENN	Border1	Border2	Safelevel	S-RSB$*$
ecoli0137vs26	0.7953	0.9678	0.9660	0.9692	0.9909	0.9909	0.9735	0.9891
shuttle0vs4	1.0000	0.9999	0.9999	1.0000	1.0000	1.0000	0.9986	0.9998
yeastB1vs7	0.7608	0.9351	0.9097	0.9376	0.9822	0.9825	0.8759	0.9505
shuttle2vs4	1.0000	0.9990	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
glass016vs2	0.8710	0.9716	0.9375	0.9595	0.9843	0.9764	0.8071	0.9629
glass016vs5	0.9843	0.9921	0.9864	0.9936	0.9943	0.9950	0.9500	0.9936
pageblocks13vs4	0.9989	0.9975	0.9975	0.9972	0.9989	0.9977	0.9868	0.9949
yeast05679vs4	0.8508	0.9526	0.9199	0.9423	0.9730	0.9730	0.8768	0.9494
yeast1289vs7	0.6290	0.9465	0.9414	0.9466	0.9879	0.9880	0.9223	0.9722
yeast1458vs7	0.5000	0.9158	0.8770	0.9038	0.9838	0.9802	0.8967	0.9680
yeast2vs4	0.9158	0.9814	0.9746	0.9787	0.9911	0.9860	0.9204	0.9687
Ecoli4	0.9430	0.9769	0.9768	0.9827	0.9941	0.9937	0.9636	0.9858
Yeast4	0.7470	0.9101	0.9136	0.9007	0.9894	0.9883	0.9434	0.9634
Vowel0	0.9999	0.9967	0.9983	0.9943	0.9979	0.9972	0.9884	0.9979
Yeast2vs8	0.5563	0.9125	0.9329	0.8960	0.9895	0.9892	0.9421	0.9862
Glass4	0.9403	0.9844	0.9845	0.9670	0.9925	0.9938	0.9397	0.9882
Glass5	0.9702	0.9976	0.9902	0.9705	0.9927	0.9945	0.9555	0.9884
Glass2	0.9350	0.9571	0.9529	0.9280	0.9841	0.9791	0.8223	0.9765
Yeast5	0.9453	0.9777	0.9802	0.9820	0.9932	0.9925	0.9799	0.9895
Yeast6	0.7762	0.9242	0.9243	0.9314	0.9918	0.9933	0.9710	0.9881
abalone19	0.5000	0.8544	0.8634	0.8851	0.8544	0.8544	0.9782	0.9902
abalone918	0.7130	0.9531	0.9314	0.9273	0.9871	0.9826	0.9421	0.9684
cleveland0vs4	0.8648	0.9893	0.9840	0.9906	0.9916	0.9916	0.9185	0.9817
ecoli01vs235	0.9097	0.9693	0.9801	0.9735	0.9858	0.9881	0.9114	0.9761
ecoli01vs5	0.9028	0.9727	0.9797	0.9820	0.9847	0.9875	0.9318	0.9409
ecoli0146vs5	0.9178	0.9861	0.9887	0.9872	0.9928	0.9913	0.9514	0.9824
ecoli0147vs2356	0.8578	0.9752	0.9757	0.9690	0.8378	0.9862	0.9328	0.9821
ecoli0147vs56	0.8842	0.9821	0.9915	0.9903	0.9902	0.9906	0.9475	0.9890
ecoli0234vs5	0.9313	0.9828	0.9845	0.9805	0.9931	0.9911	0.9368	0.9801
ecoli0267vs35	0.8788	0.9758	0.9823	0.9842	0.9839	0.9882	0.8991	0.9864
ecoli034vs5	0.9188	0.9792	0.9845	0.9818	0.9903	0.9896	0.9340	0.9854
ecoli0346vs5	0.9118	0.9851	0.9850	0.9850	0.9878	0.9892	0.9439	0.9770
ecoli0347vs56	0.8600	0.9709	0.9807	0.9786	0.9871	0.9860	0.9369	0.9822
ecoli046vs5	0.9368	0.9843	0.9889	0.9881	0.9877	0.9891	0.9228	0.9816
ecoli067vs35	0.8573	0.9469	0.9513	0.9488	0.9763	0.9706	0.8888	0.9631
ecoli067vs5	0.9363	0.9819	0.9883	0.9901	0.9875	0.9894	0.9044	0.9869
glass0146vs2	0.7879	0.9661	0.9734	0.9684	0.9794	0.9794	0.8378	0.9688
glass015vs2	0.8910	0.9605	0.9743	0.9682	0.9839	0.9806	0.8210	0.9387
glass04vs5	0.9940	0.9895	0.9892	0.9892	0.9940	0.9970	0.9382	0.9879
glass06vs5	0.9950	0.9861	0.9884	0.9871	0.9950	0.9962	0.9280	0.9937
led7digit02456789vs1	0.9022	0.9708	0.9810	0.9965	0.9708	0.9708	0.9095	0.9729
yeast0359vs78	0.7028	0.9444	0.9560	0.9594	0.9635	0.9671	0.8481	0.9386
yeast0256vs3789	0.7563	0.9474	0.9742	0.9717	0.9736	0.9740	0.8843	0.9631
yeast02579vs368	0.8809	0.9843	0.9890	0.9862	0.9874	0.9874	0.9420	0.9851
Mean	0.8593	0.9667	0.9666	0.9670	0.9813	0.9843	0.9251	0.9776

Tabla 2.4: Comparación del AUC alcanzado por la propuesta y 6 algoritmos de preprocesamiento en test

Dataset	Original	Smote	S-TL	S-ENN	Border1	Border2	Safelevel	S-RSB∗
ecoli0137vs26	0.7481	0.8136	0.8136	0.8209	**0.8445**	**0.8445**	0.8118	**0.8445**
shuttle0vs4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9988	**1.0000**
yeastB1vs7	0.6275	0.7003	0.7371	0.7277	0.6422	0.6407	0.6621	**0.8617**
shuttle2vs4	**1.0000**	0.9917	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
glass016vs2	0.5938	0.6062	0.6388	**0.6390**	0.5738	0.5212	0.6338	0.6376
glass016vs5	**0.8943**	0.8129	0.8629	0.8743	0.8386	0.8300	0.8429	0.8800
pageblocks13vs4	**0.9978**	0.9955	0.9910	0.9888	**0.9978**	0.9944	0.9831	0.9978
yeast05679vs4	0.6802	0.7602	0.7802	0.7569	0.7473	0.7331	**0.7825**	0.7719
yeast1289vs7	0.6156	0.6832	0.6332	0.7037	0.6058	0.5473	0.5603	**0.7487**
yeast1458vs7	0.5000	0.5367	0.5563	0.5201	0.4955	0.4910	0.5891	**0.6183**
yeast2vs4	0.8307	0.8588	0.9042	0.9153	0.8635	0.8576	0.8647	**0.9681**
Ecoli4	0.8437	0.8310	0.8544	**0.9044**	0.8358	0.8155	0.8386	0.8544
Yeast4	0.6135	0.7004	0.7307	0.7257	0.7124	0.6882	**0.7945**	0.7609
Vowel0	0.9706	0.9494	0.9444	0.9455	0.9278	**0.9766**	0.9566	0.9678
Yeast2vs8	0.5250	0.8066	0.8045	**0.8197**	0.6827	0.6968	0.8112	0.7370
Glass4	0.7542	0.8508	**0.9150**	0.8650	0.7900	0.8325	0.9020	0.8768
Glass5	0.8976	0.8829	0.8805	0.7756	0.8854	0.8402	0.8939	**0.9232**
Glass2	0.7194	0.5424	0.6269	0.7457	0.7092	0.5701	0.6979	**0.7912**
Yeast5	0.8833	0.9233	0.9427	0.9406	0.9118	0.9219	0.9542	**0.9622**
Yeast6	0.7115	0.8280	**0.8287**	0.8270	0.7928	0.7485	0.8163	0.8208
abalone19	0.5000	0.5202	0.5162	0.5166	0.5202	0.5202	**0.5363**	0.5244
abalone918	0.5983	0.6215	0.6675	0.7193	0.7216	0.6819	**0.8112**	0.6791
cleveland0vs4	0.6878	0.7908	0.8376	0.7605	0.7194	0.7255	**0.8511**	0.7620
ecoli01vs235	0.7136	0.8377	**0.8495**	0.8332	0.7377	0.7514	0.7550	0.7777
ecoli01vs5	0.8159	0.7977	0.8432	0.8250	0.8318	0.8295	**0.8568**	0.7818
ecoli0146vs5	0.7885	**0.8981**	**0.8981**	**0.8981**	0.7558	0.8058	0.8519	0.8231
ecoli0147vs2356	0.8051	0.8277	0.8195	0.8228	0.7465	**0.8320**	0.8149	0.8154
ecoli0147vs56	0.8318	0.8592	0.8424	0.8424	0.8420	0.8453	0.8197	**0.8670**
ecoli0234vs5	0.8307	0.8974	0.8920	0.8947	0.8613	0.8586	0.8700	**0.9058**
ecoli0267vs35	0.7752	0.8155	**0.8604**	0.8179	0.8352	0.8102	0.8380	0.8227
ecoli034vs5	0.8389	0.9000	0.9361	0.8806	0.8806	0.9028	0.8306	**0.9417**
ecoli0346vs5	0.8615	**0.8980**	0.8703	**0.8980**	0.8534	0.8838	0.8520	0.8649
ecoli0347vs56	0.7757	0.8568	0.8482	0.8546	0.8427	0.8449	0.7995	**0.8984**
ecoli046vs5	0.8168	0.8701	0.8674	0.8869	0.8615	0.8892	0.8923	**0.9476**
ecoli067vs35	0.8250	0.8500	0.8125	0.8125	0.8550	**0.8750**	0.7950	0.8525
ecoli067vs5	0.7675	0.8475	0.8425	0.8450	0.8875	**0.8900**	0.7975	0.8800
glass0146vs2	0.6616	0.7842	0.7454	0.7095	0.6565	0.6958	0.7465	**0.7978**
glass015vs2	0.5011	0.6772	0.7040	**0.7957**	0.5196	0.5817	0.7215	0.7065
glass04vs5	0.9941	0.9816	0.9754	0.9754	0.9941	**1.0000**	0.9261	0.9941
glass06vs5	**0.9950**	0.9147	0.9597	0.9647	**0.9950**	0.9000	0.9137	0.9650
led7digit02456789vs1	0.8788	0.8908	0.8822	0.8379	0.8908	0.8908	**0.9023**	0.9019
yeast0359vs78	0.5868	0.7047	0.7214	0.7024	0.6228	0.6438	0.7296	**0.7400**
yeast0256vs3789	0.6606	**0.7951**	0.7499	0.7817	0.7528	0.7644	0.7551	0.7857
yeast02579vs368	0.8432	**0.9143**	0.9007	0.9138	0.8810	0.8901	0.9003	0.9105
Mean	0.7673	0.8142	0.8247	0.8247	0.7937	0.7923	0.8173	**0.8402**

Tabla 2.5: Posición ordenada de algoritmo ganador

	Original	**Smote**	**S-TL**	**S-ENN**	**Border1**	**Border2**	**safelevel**	**S-RSB∗**	**Total**
Training	2/2	1/0	3/1	2/2	16/4	15/4	0/1	1/1	**40/4 empates**
Test	1/3	3/1	4/1	4/2	0/4	5/2	7/1	15/3	**39/5 empates**

Ganador absoluto/empate

veces ese primer lugar es compartido con otros algoritmos (empate). La última columna también contiene 2 números, el primero indica cuántas veces el mejor resultado den AUC fue alcanzado por un único algoritmo, el segundo cuando este lugar fue compartido por varios algoritmos.

La tabla 2.6 muestra la posición ordenada de los algoritmos para cada conjunto de datos. Como se puede observar $SMOTE - RSB^*$ aparece 18 veces en primer lugar, 7 veces en segundo, 9 veces es tercero y solo 1 vez en las últimas 3 posiciones.

Para encontrar el mejor algoritmo de preprocesamiento, se utilizó una prueba de múltiples comparaciones. En la tabla 2.7 se puede observar que el mejor ranking es obtenido por el algoritmo que se propone en esta tesis, las dos últimas posiciones corresponden a los algoritmos Borderline-SMOTE1 y Borderline-SMOTE2.

Finalmente se realiza una prueba de Iman-Davenport (con una F-distribución de 6 y 258 grados de libertad para $Nds = 44$) con el objetivo de encontrar diferencias estadísticas entre los algoritmos, obteniéndose p-value cercano a cero. La tabla 6 muestra los resultados del procedimiento Holm comparando $SMOTE - RSB^*$con el resto de los algoritmos. Los algoritmos han sido ordenados por el orden del $z - value$ obtenido. Además, usando una distribución normal, se obtiene el correspondiente $p - value$ asociado a cada comparación y éste puede ser comparado con el α/i que se encuentra en la misma fila de la tabla, para mostrar cuál de las hipótesis asociadas puede ser rechazada a favor del algoritmo de mejor ranking ($SMOTE - RSB^*$). Como se puede observar la hipótesis es rechazada en todos los casos, demostrándose que la propuesta que se realiza es estadísticamente superior a todos los algoritmos comparados.

Tabla 2.6: Posición ordenada para cada conjunto de datos en test

Datasets	**1**st	**2**nd	**3**rd	**4**th	**5**th	**6**th	**7**th	**8**th
ecoli0137vs26	S-RSB$^{a}_{*}$	Border2^{a}	Border1^{a}	S-ENN	S-TL	Smote	safelevel	Original
shuttle0vs4	S-RSB$_{*}$	Border2	Border1	S-ENN	S-TL	Smote	Original	safelevel
yeastB1vs7	S-RSB$_{*}$	S-TL	S-ENN	Smote	safelevel	Border1	Border2	Original
shuttle2vs4	S-RSB$^{a}_{*}$	safelevel^{a}	Border2^{a}	Border1^{a}	S-ENN^{a}	S-TL^{a}	Original^{a}	Smote
glass016vs2	S-ENN	S-TL	S-RSB$_{*}$	safelevel	Smote	Original	Border1	Border2
glass016vs5	Original	S-RSB$*$	S-ENN	S-TL	safelevel	Border1	Border2	Smote
pageblocks13vs4	S-RSB$^{a}_{*}$	Original^{a}	Border1^{a}	Smote	Border2	S-TL	S-ENN	safelevel
yeast05679vs4	safelevel	S-TL	S-RSB$_{*}$	Smote	S-ENN	Border1	Border2	Original
yeast1289vs7	S-RSB$_{*}$	S-ENN	Smote	S-TL	Original	Border1	safelevel	Border2
yeast1458vs7	S-RSB$_{*}$	safelevel	S-TL	Smote	S-ENN	Original	Border1	Border2
yeast2vs4	S-RSB$_{*}$	S-ENN	S-TL	safelevel	Border1	Smote	Border2	Original
Ecoli4	S-ENN	S-RSB$_{*}$	S-TL	Original	safelevel	Border1	Smote	Border2
Yeast4	safelevel	S-RSB$_{*}$	S-TL	S-ENN	Border1	Smote	Border2	Original
Vowel0	Border2	Original	S-RSB$_{*}$	safelevel	Smote	S-ENN	S-TL	Border1
Yeast2vs8	S-ENN	safelevel	Smote	S-TL	S-RSB$_{*}$	Border2	Border1	Original
Glass4	S-TL	safelevel	S-RSB$_{*}$	S-ENN	Smote	Border2	Border1	Original
Glass5	S-RSB$_{*}$	Original	safelevel	Border1	Smote	S-TL	Border2	S-ENN
Glass2	S-RSB$_{*}$	S-ENN	Original	Border1	safelevel	S-TL	Border2	Smote
Yeast5	S-RSB$_{*}$	safelevel	S-TL	S-ENN	Smote	Border2	Border1	Original
Yeast6	S-TL	Smote	S-ENN	S-RSB$_{*}$	safelevel	Border1	Border2	Original
abalone19	safelevel	S-RSB$_{*}$	Border2	Border1	Smote	S-ENN	S-TL	Original
abalone918	safelevel	Border1	S-ENN	Border2	S-RSB$_{*}$	S-TL	Smote	Original
cleveland0vs4	safelevel	S-TL	Smote	S-RSB$_{*}$	S-ENN	Border2	Border1	Original
ecoli01vs235	S-TL	Smote	S-ENN	S-RSB$_{*}$	safelevel	Border2	Border1	Original
ecoli01vs5	safelevel	S-TL	Border1	Border2	S-ENN	Original	Smote	S-RSB$_{*}$
ecoli0146vs5	Smote	S-ENN	S-TL	safelevel	S-RSB$_{*}$	Border2	Original	Border1
ecoli0147vs2356	Border2	Smote	S-ENN	S-TL	S-RSB$_{*}$	safelevel	Original	Border1
ecoli0147vs56	S-RSB$_{*}$	Smote	Border2	S-ENN	S-TL	Border1	Original	safelevel
ecoli0234vs5	S-RSB$_{*}$	Smote	S-ENN	S-TL	safelevel	Border1	Border2	Original
ecoli0267vs35	S-TL	safelevel	Border1	S-RSB$_{*}$	S-ENN	Smote	Border2	Original
ecoli034vs5	S-RSB$_{*}$	S-TL	Border2	Smote	Border1	S-ENN	Original	safelevel
ecoli0346vs5	S-ENN^{a}	Smote^{a}	Border2	S-TL	S-RSB$_{*}$	Original	Border1	safelevel
ecoli0347vs56	S-RSB$_{*}$	Smote	S-ENN	S-TL	Border2	Border1	safelevel	Original
ecoli046vs5	S-RSB$_{*}$	safelevel	Border2	S-ENN	Smote	S-TL	Border1	Original
ecoli067vs35	Border2	Border1	S-RSB$_{*}$	Smote	Original	S-ENN	S-TL	safelevel
ecoli067vs5	Border2	Border1	S-RSB$_{*}$	Smote	S-ENN	S-TL	safelevel	Original
glass0146vs2	S-RSB$_{*}$	Smote	safelevel	S-TL	S-ENN	Border2	Original	Border1
glass015vs2	S-ENN	safelevel	S-RSB$_{*}$	S-TL	Smote	Border2	Border1	Original
glass04vs5	Border2	S-RSB$_{*}$	Border1	Original	Smote	S-ENN	S-TL	safelevel
glass06vs5	Border1^{a}	Original^{a}	S-RSB$_{*}$	S-ENN	S-TL	Smote	safelevel	Border2
led7digit02456789vs1	safelevel	S-RSB$_{*}$	Border2	Border1	Smote	S-TL	Original	S-ENN
yeast0359vs78	S-RSB$_{*}$	safelevel	S-TL	Smote	S-ENN	Border2	Border1	Original
yeast0256vs3789	Smote	S-RSB$_{*}$	S-ENN	Border2	safelevel	Border1	S-TL	Original
yeast02579vs368	Smote	S-ENN	S-RSB$_{*}$	S-TL	safelevel	Border2	Border1	Original

Tabla 2.7: Ranking obtenido con una prueba de Friedman

Algorithm	Ranking
S-RSB$*$	2.61364
S-ENN	3.92045
S-TL	3.96591
Smote	4.23864
Safelevel	4.34091
Boderline-SMOTE2	5.18182
Boderline-SMOTE1	5.36364

Tabla 2.8: Tabla de Holm para $\alpha = 0{,}05$ usando $SMOTE - RSB^*$ como muestra de control

i	Algorithm	$z = (R_0 - R_i)/SE$	p	Holm/Hochberg/Hommel	Hypothesis
6	Borderline-SMOTE1	5.2658490926	1.395E-7	0.008333	Reject
5	Borderline-SMOTE2	4.9176937807	8.756E-7	0.01	Reject
4	Safelevel	3.3074754631	9.414E-4	0.0125	Reject
3	Smote	3.1116381002	0.001860	0.016667	Reject
2	S-TL	2.5894051323	0.009614	0.025	Reject
1	S-ENN	2.5023663043	0.012336	0.05	Reject

2.5. Conclusiones parciales

En este capítulo se ha presentado un nuevo algoritmo para editar conjuntos de entrenamiento altamente desbalanceados. La propuesta está enmarcada dentro del conjunto de técnicas conocidas como híbridas, es decir que mezclan sobremuestreo y bajomuestreo.

La principal novedad de esta propuesta es que la calidad de los ejemplos sintéticos creados por SMOTE es evaluada haciendo uso de la TCA, específicamente el concepto de aproximación inferior dado por la teoría. Esta evaluación de instancias nos permite seleccionar solamente aquellas instancias artificiales que con absoluta certeza pertenecen a la clase minoritaria, pues están contenidas en su aproximación inferior.

Con el resultado observado en el estudio experimental, queda evidenciada el buen comportamiento del algoritmo al lograr mejores resultados que otros 6 algoritmos reconocidos del estado del arte.

CAPÍTULO 3

Un algoritmo de preprocesamiento basado en SMOTE y en Teoría de los Conjuntos Aproximados Difusos como algoritmo de limpieza para ejemplos artificiales y originales de la clase mayoritaria

"Knowing is not enough; we most apply.Willing is not enough; we must do."

– Johann W. von Goethe.

En este capítulo se introduce un nuevo algoritmo híbrido de preprocesamiento para editar conjuntos de datos de alto desbalance. El algoritmo propuesto genera ejemplos sintéticos usando SMOTE y luego aplica un algoritmo de limpieza basado la Teoría de los Conjuntos Aproximados Difusos (TCAD) para eliminar aquellas instancias sintéticas o de la clase mayoritaria que tengan una pertenencia muy baja a la aproximación inferior de su clase. Se proponen 2 formas diferentes de fijar el umbral para realizar la limpieza: la

primera consiste en usar el mismo umbral para los ejemplos artificiales y los originales de la clase mayoritaria, y la segunda consiste en usar umbrales distintos para ambas ediciones.

Como se estudió en la sección 3.1.1 el enfoque difuso de la TCA ha abierto nuevas puertas en la edición de conjuntos de entrenamiento Jensen and Cornelis (2010). La TCAD ha sido usada para evaluar la pertenencia de los ejemplos a cada una de las regiones. Su uso supone ciertas ventajas por encima de la TCA clásica, debido a que se puede flexibilizar el rigor de la evaluación a través del valor de umbral usado.

En la TCA clásica solo podemos saber si un objeto pertenece o no a la región positiva de una clase, y esa respuesta siempre es booleana. Sin embargo la TCAD cuantifica el grado en el que los objetos pertenecen a cada una de las regiones; lo cual indica que es posible suavizar la frontera de decisión.

Los objetos con grado de pertenencia cercano a 1 serán buenos representantes de su clase, por el contrario los objetos con grado cercano a cero podrán ser considerados como objetos ruidosos o muy alejados de su clase (más cercanos a la contraria). Partiendo de estas ideas en este capítulo se diseñan dos nuevos algoritmos de preprocesamieno híbridos para conjuntos altamente desbalanceados. El primero de estos algoritmos edita usando el mismo umbral los ejemplos originales de la clase mayoritaria y los sintéticos creados por SMOTE y el segundo usa umbrales distintos para ambas ediciones.

En las siguientes subsecciones se expondrán los principales conceptos y definiciones relacionados con la TCAD necesarios para el desarrollo de nuestras propuestas, se describirán en detalle los dos algoritmos propuestos, la realización del ajuste de parámetros, se describirán los conjuntos de datos utilizados, así como el estudio experimental que sustentará nuestras hipótesis. Se describirá además la aplicación de uno de los algoritmos en la solución de un problema de la ingeniería eléctrica: el diagnóstico de la necesidad de mantenimiento de interruptores de alta potencia.

3.1. SMOTE-FRST: un algoritmo de preprocesamiento que combina SMOTE y la Teoría de los Conjuntos Aproximados Difusos

En este epígrafe se introduce el algoritmo SMOTE-FRST, un nuevo algoritmo híbrido de edición para conjuntos de datos no balanceados.

Este algoritmo consta de dos estados fundamentales, los cuales se repiten hasta que el conjunto quede balanceado o hasta que un número máximo de iteraciones sean ejecutadas:

1. Aplicar SMOTE para balancear el conjunto de entrenamiento, creando instancias minoritarias sintéticas.
2. Eliminar las instancias sintéticas, o las mayoritarias, cuyo grado de pertenencia a la región positiva usando la TCAD sea menor que un umbral dado.

Nuestra hipótesis es que algunas de las instancias sintéticas introducidas pueden no ser apropiadas para ser usadas en la fase de aprendizaje, y por tanto deben ser eliminadas. De forma similar, algunas de las instancias mayoritarias del conjunto original que no pertenezcan a la región fuzzy-rough positiva de su clase, también serán eliminadas. Nuestro procedimiento no se les aplica a las instancias positivas originales, dado que son muy pocas y además están relativamente dispersas, por lo que es mejor no tocarlas.

3.1.1. Un enfoque difuso de la Teoría de los Conjuntos Aproximados

Los conjuntos difusos Zadeh (1965), así como los conjuntos aproximados Pawlak (1982), han marcado significativamente la forma en la que se opera con sistemas con información imperfecta en la actualidad. Ambos han permitido el surgimiento de amplias comunidades de investigación que han marcado un hito tanto al nivel teórico como en las aplicaciones. Aunque es claro que las dos teorías son complementarias más que competitivas, la existencia de notorias similitudes entre ambos conceptos ha desencadenado la creación de una teoría híbrida que combina el esfuerzo de ambos.

En la TCA clásica la separabilidad es vista como un modelo en blanco y negro (las instancias son separables o no), lo cual limita en cierta forma la aplicación de la teoría pues en la práctica la separabilidad ocurre en forma gradual.

Es por esta razón que la TCA ha sido hibridada con los conjuntos difusos Zadeh (1965) (fuzzy sets), los cuales modelan una transacción gradual de un concepto. En la teoría TCAD resultante, las relaciones de inseparabilidad son modeladas por medio de relaciones difusas simétricas y reflexivas. En esta investigación se asumirán las definiciones que se enuncian a continuación, las cuales también fueron usadas en Ramentol et al. (2012).

Sea X un conjuntos de instancias y $\mathscr{A}$ un conjunto de atributos. Dado un atributo continuo a en $\mathscr{A}$, y dos instancias x e y en X, tal que $a(x)$ y $a(y)$ representan el valor de las instancias x e y en el atributo a respectivamente, se define:

$$R_a(a,b) = max(1 - \frac{\mid a(y) - a(x) \mid}{\sigma_a}, 0) \tag{3.1}$$

donde σ_a es la desviación estándar del atributo a. $R_a(a,b)$ es un valor entre 0 y 1 que cuantifica la separabilidad entre x e y : mientras más alto sea este valor, más cercanos (más similares) serán x e y. Por otra parte si a es un atributo nominal, se define:

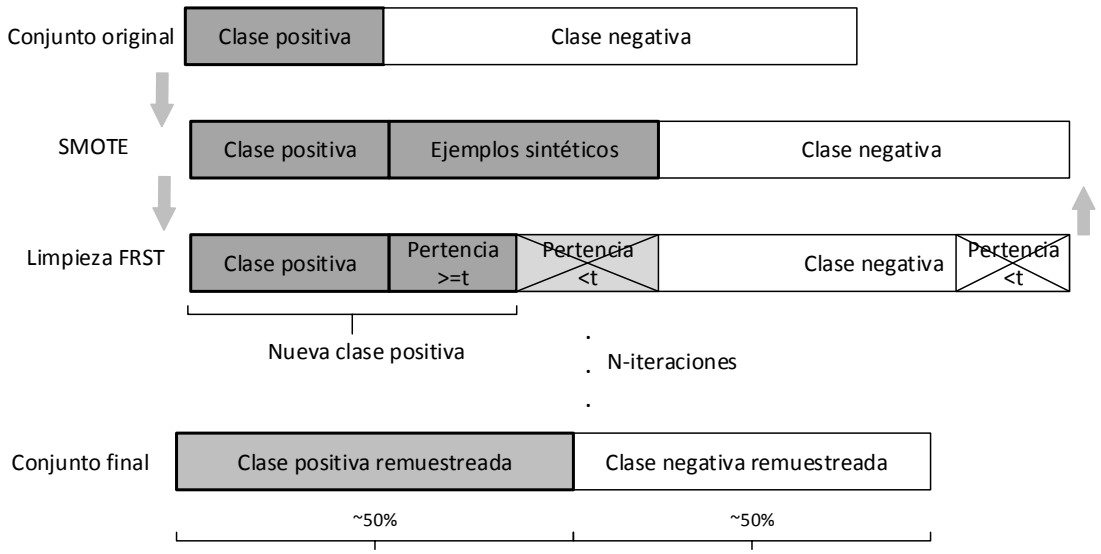
$$R_a(x,y) = \begin{cases} 1 & if\ a(x)=a(y) \\ 0 & otherwise \end{cases} \tag{3.2}$$

En este caso, la inseparabilidad con respecto a a es de 2 valores, si dos instancias tienen valores diferentes para a, éstas pueden ser separables ($R_a(x,y) = 0$), mientras que en el caso contrario ($R_a(x,y) = 1$), no pueden. Con el objetivo de comparar x e y usando el todos los atributos en $\mathscr{A}$, se calcula el siguiente valor agregado:

$$R(x,y) = T_L \underbrace{(R_a(x,y))} \tag{3.3}$$

donde T_L representa la t-norma de *Lukasiewicz* dada por $T_L(v_1,\ldots,v_n) = max(0, v_1 + \ldots + v_n - n)$ para $v_1 \ldots, v_n \in [0,1]$, donde $n = |\mathscr{A}|$. R es llamada la relación de inseparabilidad difusa aproximada (fuzzy-rough indiscernibility relation).

Figura 3.1: Esquema del funcionamiento de SMOTE-TCAD



Usando a R, es posible evaluar para cada instancia x, su grado de pertenencia a la región positiva POS, a través de encontrar su instancia más similar y la cual tiene una etiqueta de clase diferente:

$$POS(x) = \min_{class(y) \neq class(x)} 1 - R(x,y) \tag{3.4}$$

La idea de la región positiva es que las instancias x en la frontera de una clase (por ejemplo si existe una instancia similar en otra clase) tendrá un valor $POS(x)$ pequeño comparado con las instancias que se encuentran en el centro de la clase. Esto es lo que hace a la región positiva una métrica muy eficiente para evaluar la "calidad" de las instancias, es decir, si son instancias representativas para su clase.

3.1.2. Algoritmo SMOTE-FRST

La figura muestra un esquema del funcionamiento el algoritmo SMOTE-FRST.

El algoritmo de limpieza que se propone está basado en el uso de la TCAD, el mismo está conformado por los siguientes pasos:

1. "Paso 1": usar SMOTE para balancear el conjunto de datos original (este paso se corresponde con el estado 1).
2. "Paso 2": construir el conjunto resultante (salida del algoritmo) incluyendo solo los ejemplos originales de la clase minoritaria.
3. "Paso 3": insertar en el *resultSet* todos aquellos ejemplos de la clase mayoritaria que tengan un grado pertenecia a la aproximación inferior de su clase por encima de un umbral dado como parámetro (*threshold*).
4. "Paso 4": insertar en el *resultSet* todos aquellos ejemplos sintéticos que tengan un grado pertenencia a la aproximación inferior de su clase por encima de un umbral dado como parámetro (*threshold*).
5. "Paso 5": Si los ejemplos encontrados en el paso 4 no alcanzan para balancear el conjunto y no se ha alcanzado un número máximo de iteraciones, entonces se comienza por el paso 1.

La figura 3.1 muestra el algoritmo asociado a los pasos descritos previamente.

3.2. Marco experimental: parámetros, resultados y pruebas estadísticas

Para el estudio experimental se utilizaron los 44 conjuntos de alto desbalance del repositorio de la UCI Murphy and Aha (1994) que aparecen en la tabla 2.1. El clasificador seleccionado es el árbol de decisión C4.5 Quinlan (1993) debido a que es un clasificador bastante robusto frente a conjuntos no balanceados. El algoritmo se comparó con otros 6 algoritmos de remuestreo, que fueron descritos en el epígrafe 1.3.1

- EL algoritmo SMOTE solo, sin ningún algoritmo de limpieza, Chawla et al. (2002).
- SMOTE con un algoritmo de limpieza basado en Tomek Links (S-TL), Batista et al. (2004).

Algoritmo 3.1 Algoritmo propuesto SMOTE-TCAD

Entrada: Threshold $threshold$
iteration number n_i
SMOTE parameters
$neg[]$ array of negatives examples $pos[]$ array of positives examples
Salida: $resultSet$

1: Step1: Using SMOTE we create a set of synthetic data ($syntheticInstances[]$) for the minority class until the training set is balanced.
2: add $pos[]$ to $resultSet$
3: $executionNumber = 0$
4: $isbalance = false$
5: **while** $(executionNumber <= n_i)$ & $!(isbalance)$ **do**
6: **for** $i \leftarrow 1$ **to** $(syntheticInstances[].lenght)$ **do**
7: Compute $fuzzyLowerMembership(syntheticInstances[i])$
8: **if** $fuzzyLowerMembership[i] >= threshold$ **then**
9: insert $syntheticInstances[i]$ in $resultSet$
10: $nsynt++$
11: **end if**
12: **end for**
13: **for** $j \leftarrow 1$ **to** $neg[]$ **do**
14: Compute $fuzzyLowerMembership(neg[i])$
15: **if** $fuzzyLowerMembership[i] >= threshold$ **then**
16: insert $nneg[i]$ in $resultSet$
17: $nneg++$
18: **end if**
19: **end for**
20: $balance = (nneg = (nsynt + pos[].lenght))$
21: $executionNumber++$
22: **end while**
23: Step5: If there are no instances in the lower approximation, the solution is given as the set balanced with SMOTE, all synthetic instances are include in $resultSet$.

- SMOTE con un algoritmo de limpieza que usa la técnica de la edición del vecino más cercano (S-ENN), Batista et al. (2004).
- Borderline-SMOTE, donde solo las instancias en la frontera son remuestreadas (BORDER1), Han et al. (2005).
- Una variación a SMOTE-BL1, pero que genera instancias sintéticas cercanas a la clase minoritaria (BORDER2), Han et al. (2005).
- SMOTE ponderando las instancias minoritarias atendiendo a su nivel de "seguridad" (SAFELEVEL), Bunkhumpornpat et al. (2009).

3.2.1. Parámetros

En los experimentos se usaron los siguientes parámetros para el algoritmo SMOTE-TCAD:

- k: número de vecinos más cercanos, fijado en 5.
- Función de distancia para obtener los vecinos más cercanos, se usó la distancia de Euclidean
- Distribución por clases para el remuestreo, se definió 50 %-50 %.
- Umbral para determinar la pertenencia a la aproximación inferior, se definió 0.8.

Para el árbol de decisión C4.5 se usaron los parámetros definidos por defecto en KEEL Alcalá et al. (2009).

- pruned = TRUE
- confidence level: 0.25
- instancesPerLeaf = 2

Los datos fueron particionados en 5 para la validación cruzada (5-fold cross validation), de forma tal que la distribución de ejemplos por clases quedara uniforme.

Para los algoritmos del estado del arte los parámetros usados se muestran en la tabla 3.1:

Tabla 3.1: Parámetros usados para los algoritmos del estado del arte

Algoritmos	Parámetros
SMOTE	Number of Neighbors = 5 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function = HVDM Type of Interpolation = standard
SMOTE-TomekLinks	Number of Neighbors = 5 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function = HVDM
SMOTE-ENN	Number of Neighbors ENN = 3 Number of Neighbors SMOTE = 5 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function (SMOTE) = HVDM Distance Function (ENN) = Euclidean
Borderline-SMOTE1	Number of Neighbors SMOTE = 5 Number of Neighbors for considering a instance border = 3 Type of borderline SMOTE = 1 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function (SMOTE) = HVDM Type of Interpolation = standard
Borderline-SMOTE2	Number of Neighbors SMOTE = 5 Number of Neighbors for considering a instance border = 3 Type of borderline SMOTE = 2 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function (SMOTE) = HVDM Type of Interpolation = standard
Safe-Level-SMOTE	Number of Neighbors = 5 Type of SMOTE = both Balancing = YES Quantity of generated examples = 1 Distance Function = HVDM Type of Interpolation = standard

3.2.2. Análisis comparativo

Se comparará el algoritmo $SMOTE-FRST$ con otros 6 algoritmos de preprocesamiento para conjuntos no balanceados, todos éstos basados en SMOTE, los cuales fueron explicados detalles en el epígrafe 1.3.1. La comparación se realizará con el propio SMOTE y con 5 hibridaciones de éste con algoritmos de limpieza: SMOTE-TomekLinks, SMOTE-ENN, Borderline-SMOTE1, Borderline-SMOTE2 y Safe-Level-SMOTE.

En las tablas 3.2 y 3.3 se muestran los resultados del experimental en cuanto a training y test respectivamente.

Tabla 3.2: AUC obtenido por C4.5 en training

DATASETS	SMOTE	S-TL	S-ENN	BORDER1	BORDER2	SAFELEVEL	S-TCAD
yeast-2_vs_4	0.9735	0.9803	0.9828	0.9900	0.9889	0.9174	0.9805
yeast-0-5-6-7-9_vs_4	0.9497	0.9674	0.9640	0.9722	0.9644	0.8787	0.9758
vowel0	0.9968	0.9534	0.9967	0.9978	0.9979	0.9872	0.9979
glass-0-1-6_vs_2	0.9421	0.9747	0.9751	0.9864	0.9786	0.8679	0.9853
glass2	0.9670	0.9856	0.9783	0.9867	0.9791	0.8909	0.9900
ecoli4	0.9814	0.8465	0.9861	0.9949	0.9917	0.9644	0.9907
yeast-1_vs_7	0.9458	0.9564	0.9424	0.9846	0.9790	0.8654	0.9680
shuttle-c0-vs-c4	0.9991	0.9995	0.9993	1.0000	1.0000	0.9955	0.9999
glass4	0.9751	0.9897	0.9866	0.9913	0.9932	0.9353	0.9845
page-blocks-1-3_vs_4	0.9873	0.9931	0.9960	0.9994	0.9972	0.9637	0.9947
abalone9-18	0.9670	0.9693	0.9585	0.9846	0.9842	0.8723	0.9670
glass-0-1-6_vs_5	0.9886	0.9877	0.9898	0.9936	0.9900	0.9464	0.9896
shuttle-c2-vs-c4	0.9807	0.9990	0.9979	1.0000	1.0000	0.9431	0.9967
yeast-1-4-5-8_vs_7	0.9534	0.9698	0.9610	0.9811	0.9830	0.8948	0.9798
glass5	0.9902	0.9913	0.9864	0.9933	0.9939	0.9537	0.9898
yeast-2_vs_8	0.9840	0.9829	0.9877	0.9894	0.9862	0.9445	0.9948
yeast4	0.9789	0.9805	0.9797	0.9908	0.9871	0.9395	0.9912
yeast-1-2-8-9_vs_7	0.9749	0.9747	0.9738	0.9850	0.9854	0.9213	0.9823
yeast5	0.9881	0.9937	0.9919	0.9948	0.9942	0.9799	0.9973
ecoli-0-1-3-7_vs_2-6	0.9845	0.9898	0.9908	0.9909	0.9909	0.9708	0.9906
yeast6	0.9832	0.9869	0.9875	0.9925	0.9920	0.9702	0.9952
abalone19	0.9871	0.9893	0.9891	0.9949	0.9961	0.9791	0.9876
cleveland-0_vs_4	0.9886	0.9937	0.9882	0.9878	0.9870	0.9185	0.9889
ecoli-0-1_vs_2-3-5	0.9773	0.9831	0.9797	0.9864	0.9852	0.9074	0.9803
ecoli-0-1_vs_5	0.9881	0.9925	0.9912	0.9977	0.9932	0.9460	0.9952
ecoli-0-1-4-6_vs_5	0.9851	0.9920	0.9930	0.9933	0.9899	0.9481	0.9908
ecoli-0-1-4-7_vs_2-3-5-6	0.9792	0.9870	0.9806	0.9862	0.9849	0.9296	0.9888
ecoli-0-1-4-7_vs_5-6	0.9845	0.9940	0.9923	0.9919	0.9927	0.9401	0.9935
ecoli-0-2-3-4_vs_5	0.9808	0.9894	0.9798	0.9883	0.9897	0.9430	0.9879
ecoli-0-2-6-7_vs_3-5	0.9709	0.9713	0.9748	0.9839	0.9833	0.8892	0.9887
ecoli-0-3-4_vs_5	0.9826	0.9908	0.9867	0.9903	0.9889	0.9451	0.9913
ecoli-0-3-4-6_vs_5	0.9831	0.9869	0.9829	0.9905	0.9872	0.9480	0.9895
ecoli-0-3-4-7_vs_5-6	0.9661	0.9846	0.9813	0.9871	0.9849	0.9127	0.9839
ecoli-0-4-6_vs_5	0.9829	0.9888	0.9853	0.9904	0.9891	0.9419	0.9923
ecoli-0-6-7_vs_3-5	0.9713	0.9683	0.9638	0.9831	0.9856	0.8850	0.9864
ecoli-0-6-7_vs_5	0.9775	0.9848	0.9919	0.9869	0.9831	0.9100	0.9925
glass-0-1-4-6_vs_2	0.9608	0.9736	0.9555	0.9767	0.9867	0.8896	0.9851
glass-0-1-5_vs_2	0.9524	0.9438	0.9752	0.9847	0.9734	0.8629	0.9866
glass-0-4_vs_5	0.9895	0.9892	0.9892	0.9940	0.9970	0.9307	0.9938
glass-0-6_vs_5	0.9874	0.9910	0.9909	0.9950	1.0000	0.9166	0.9866
led7digit-0-2-4-5-6-7-8-9_vs_1	0.9763	0.9772	0.9949	0.9806	0.9797	0.9049	0.9947
yeast-0-2-5-6_vs_3-7-8-9	0.9576	0.9633	0.9680	0.9743	0.9718	0.8855	0.9746
yeast-0-2-5-7-9_vs_3-6-8	0.9801	0.9852	0.9825	0.9867	0.9880	0.9244	0.9937
yeast-0-3-5-9_vs_7-8	0.9378	0.9686	0.0158	0.9690	0.9592	0.8342	0.9743
MEAN	0.9759	0.9787	0.9601	**0.9886**	0.9871	0.9249	0.9879

Tabla 3.3: AUC obtenido por C4.5 en test

DATASETS	SMOTE	S-TL	S-ENN	BORDER1	BORDER2	SAFELEVEL	S-TCAD
yeast-2_vs_4	0.8616	0.9053	0.8698	0.8556	0.8659	0.8673	0.8635
yeast-0-5-6-7-9_vs_4	0.7436	0.7527	0.7530	0.7577	0.7478	0.7961	0.8058
vowel0	0.9778	0.9794	0.9917	0.9650	0.9722	0.9522	0.9889
glass-0-1-6_vs_2	0.6724	0.6890	0.5643	0.6290	0.5900	0.6795	0.5476
glass2	0.7252	0.7229	0.7538	0.6084	0.5169	0.7202	0.7604
ecoli4	0.8747	0.8465	0.9168	0.9187	0.8592	0.8823	0.9029
yeast-1_vs_7	0.6580	0.6781	0.7243	0.6205	0.6422	0.6659	0.7040
shuttle-c0-vs-c4	0.9994	1.0000	1.0000	0.9997	1.0000	0.9959	0.9994
glass4	0.9101	0.8052	0.9176	0.7417	0.7633	0.8061	0.9176
page-blocks-1-3_vs_4	0.9853	0.9496	0.9653	0.9544	0.9865	0.9495	0.9820
abalone9-18	0.7680	0.7406	0.7672	0.7162	0.6785	0.7531	0.7680
glass-0-1-6_vs_5	0.8771	0.9243	0.9186	0.8329	0.7271	0.8543	0.9243
shuttle-c2-vs-c4	1.0000	0.9875	1.0000	1.0000	1.0000	0.9588	1.0000
yeast-1-4-5-8_vs_7	0.5473	0.4964	0.5336	0.4941	0.5008	0.5899	0.5451
glass5	0.9634	0.8659	0.9134	0.8427	0.9378	0.8439	0.9537
yeast-2_vs_8	0.8273	0.8393	0.8023	0.7816	0.6696	0.7926	0.8327
yeast4	0.7722	0.7642	0.7883	0.6980	0.6681	0.7733	0.7675
yeast-1-2-8-9_vs_7	0.5805	0.6389	0.6042	0.5503	0.5164	0.5950	0.6198
yeast5	0.9285	0.9715	0.9507	0.9247	0.9135	0.9462	0.9531
ecoli-0-1-3-7_vs_2-6	0.8099	0.8172	0.8190	0.8445	0.8445	0.7245	0.8281
yeast6	0.8030	0.8292	0.8292	0.7792	0.7638	0.8063	0.8204
abalone19	0.5807	0.5521	0.5675	0.5082	0.5111	0.5844	0.5548
cleveland-0_vs_4	0.7801	0.7102	0.7467	0.7315	0.6968	0.8511	0.8226
ecoli-0-1_vs_2-3-5	0.7709	0.8427	0.8500	0.7536	0.7714	0.8427	0.8473
ecoli-0-1_vs_5	0.8295	0.8250	0.8750	0.7864	0.8295	0.8227	0.8295
ecoli-0-1-4-6_vs_5	0.8750	0.8904	0.9000	0.8308	0.8019	0.8519	0.9250
ecoli-0-1-4-7_vs_2-3-5-6	0.8556	0.8759	0.8559	0.8686	0.8670	0.8316	0.8793
ecoli-0-1-4-7_vs_5-6	0.8522	0.8510	0.8608	0.8205	0.8653	0.8261	0.8840
ecoli-0-2-3-4_vs_5	0.9224	0.9225	0.9169	0.8584	0.7973	0.8670	0.8892
ecoli-0-2-6-7_vs_3-5	0.8379	0.7707	0.8129	0.8154	0.8528	0.7835	0.8230
ecoli-0-3-4_vs_5	0.9139	0.8750	0.8639	0.8556	0.9278	0.8861	0.8889
ecoli-0-3-4-6_vs_5	0.9176	0.8899	0.8730	0.8838	0.8534	0.8426	0.8811
ecoli-0-3-4-7_vs_5-6	0.8347	0.8282	0.8482	0.8784	0.8406	0.7803	0.8698
ecoli-0-4-6_vs_5	0.9120	0.9146	0.8896	0.8283	0.8032	0.8895	0.8842
ecoli-0-6-7_vs_3-5	0.8575	0.8375	0.8275	0.8900	0.8400	0.8425	0.8625
ecoli-0-6-7_vs_5	0.8500	0.8025	0.8150	0.8650	0.8600	0.7600	0.8625
glass-0-1-4-6_vs_2	0.6871	0.7769	0.7314	0.6997	0.6931	0.8190	0.8337
glass-0-1-5_vs_2	0.7272	0.7411	0.6914	0.5605	0.5914	0.6831	0.6489
glass-0-4_vs_5	0.9581	0.9699	0.9699	0.9941	0.9879	0.9202	0.9699
glass-0-6_vs_5	0.9200	0.9600	0.9645	0.9950	0.9795	0.9039	0.9750
led7digit-0-2-4-5-6-7-8-9_vs_1	0.8933	0.8748	0.8598	0.8931	0.8906	0.8714	0.8626
yeast-0-2-5-6_vs_3-7-8-9	0.7674	0.7658	0.7843	0.7387	0.7481	0.7720	0.7782
yeast-0-2-5-7-9_vs_3-6-8	0.9157	0.8980	0.9218	0.8798	0.8557	0.8964	0.9085
yeast-0-3-5-9_vs_7-8	0.7411	0.6970	0.7257	0.6395	0.6207	0.7383	0.7377
MEAN	0.8292	0.8244	0.8303	0.7975	0.7875	0.8141	**0.8387**

En la figura 3.2 se muestran las medias alcanzadas en AUC por los algoritmos sobre todos los conjuntos de datos.

Como se aprecia SMOTE-TCAD tiene el mejor resultado como promedio. El resultado es ligeramente mejor que el SMOTE sin algoritmo de limpieza, por tal motivo se hace necesario comprobar si esta diferencia es significativa. Para ello se aplica una prueba de Wilcoxon Wilcoxon (1945), para comparar el algoritmo propuesto con los restantes

Figura 3.2: AUC medio alcanzado por los algoritmos sobre los 44 conjuntos de datos

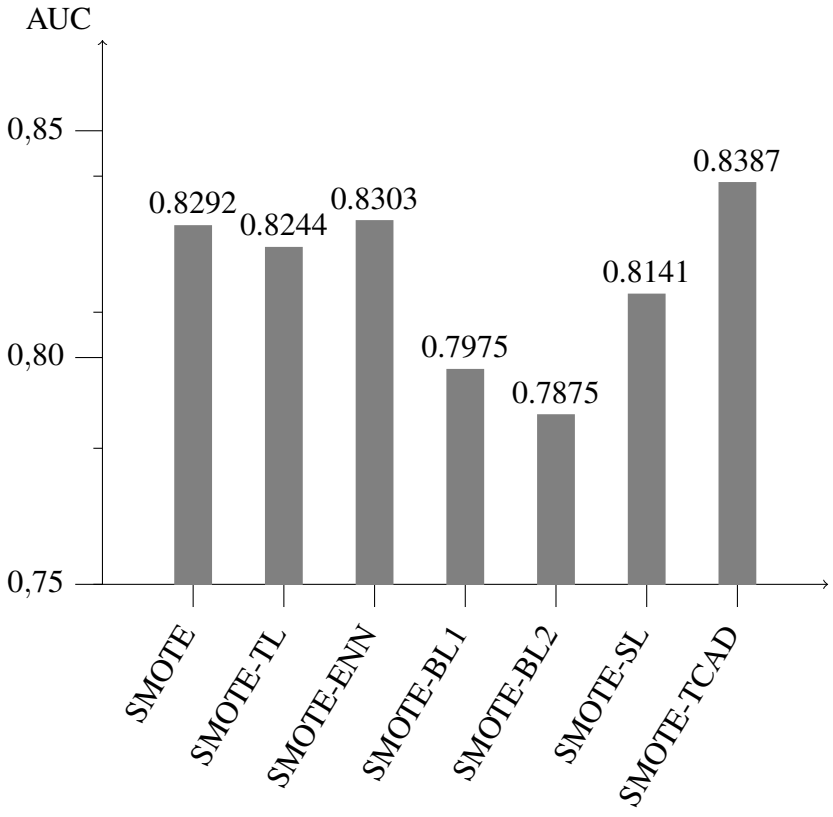


Tabla 3.4: Resultado de la prueba de Wilcoxon

VS	R^+	R^-	Exact P-value	Asymptotic P-value
SMOTE	621.0	199.0	0.003868	0.004471
TL	667.5	152.5	3.266E-4	0.000502
ENN	568.0	212.0	0.012092	0.01243
SAVELEVEL	730.0	90.0	3.488E-6	0.000016
BODER1	699.0	81.0	3.05E-6	0.000015
BORDER2	722.5	97.5	6.6339999999999995E-6	0.000025

algoritmos basados en SMOTE. La prueba de Wilcoxon es una prueba no paramétrica para 2 muestras relacionadas, que encuentra si existen diferencias significativas entre dichas muestras.

Por cada comparación se calcula la suma del ranking de la prueba de Wilcoxon a favor de SMOTE-TCAD $(R+)$, la suma del ranking a favor de los otros algoritmos es $(R-)$, así como el $p-value$ de cada comparación. Estos valores se muestran en la tabla , el $p-value$ en todos los casos está por debajo de 0.01. lo cual significa que nuestra propuesta funciona significativamente mejor que el resto de los algoritmos basados en SMOTE, con un nivel de significancia del 5 porciento.

3.3. Un algoritmo de preprocesamiento basado en SMOTE y en Teoría de los Conjuntos Aproximados Difusa con doble umbral

En este epígrafe se propone un nuevo algoritmo de preprocesamiento híbrido basado en SMOTE y en la TCAD. El algoritmo que se propone constituye una extensión del algoritmo SMOTE-FRST, solo que esta nueva propuesta es capaz de editar de forma diferente los ejemplos sintéticos y los ejemplos positivos originales.

La propuesta consta de tres estados fundamentales:

1. Aplicar SMOTE para balancear el conjunto de entrenamiento, creando instancias minoritarias sintéticas.
2. Eliminar las instancias mayoritarias que pertenezcan a la región positiva de su clase con un grado de pertenencia menor que δ_m.

3. Eliminar las instancias sintéticas que pertenezcan a la región positiva de su clase con un grado de pertenencia menor que δ_s.

En la propuesta que se introdujo en el epígrafe anterior las instancias sintéticas y las mayoritarias eran editadas usando el mismo valor de umbral, los resultados experimentales mostraron buenos resultados a favor de la propuesta. Sin embargo al presentarse una situación real (que será descrita en detalles en el siguiente capítulo), esta propuesta no obtuvo resultados satisfactorios.

El hecho es que en una aplicación real cada ejemplo sintético que se introduzca en el conjunto para ser usado en la etapa de aprendizaje, representa un ejemplo que no ha sido observado en la realidad, por lo que se debe ser muy estrictos a la hora de seleccionarlos (usar un umbral alto).

Al usar un umbral alto (si se usara el mismo umbral para las mayoritarias), se eliminarían demasiados ejemplos originales, los cuales representan observaciones de los expertos. Es por ello que define esta nueva propuesta, teniendo estas dos premisas fundamentales:

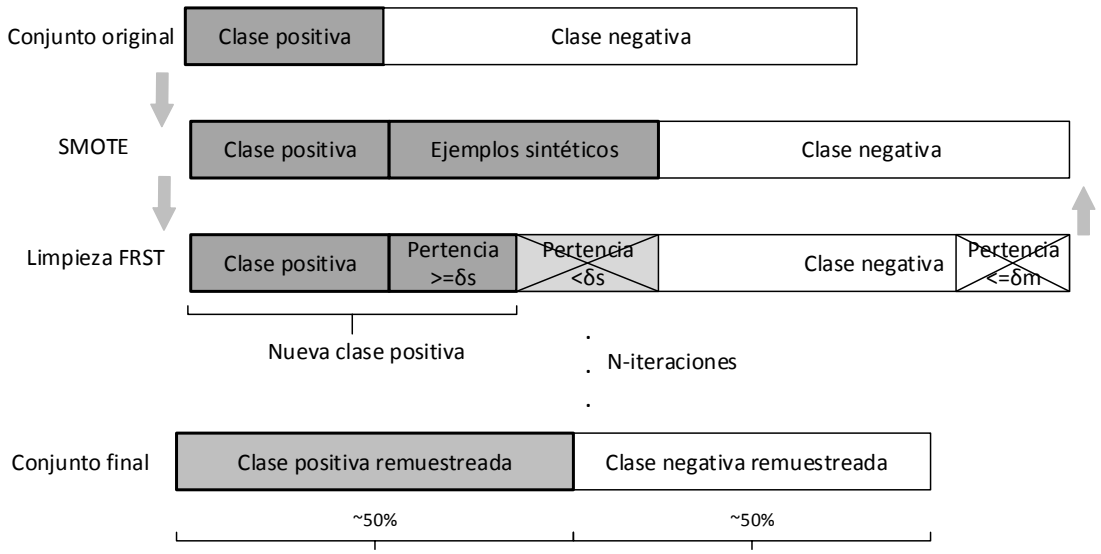
- Los ejemplos sintéticos deben tener la mayor calidad posible, se debe usar el umbral más alto, de manera que los ejemplos que se inserten en el conjunto final sean totalmente inseparables de los objetos de la clase contraria.
- Se deben eliminar aquellos ejemplos de la clase mayoritaria que por su pertenencia tan baja a la región positiva, afecten la separabilidad de las clases en la etapa de clasificación (usar un umbral muy bajo).

La figura 3.2 muestra el algoritmo asociado a los pasos descritos previamente.

El algoritmo de limpieza que se propone está basado en el uso de la TCAD, el mismo está conformado por los siguientes pasos:

1. “Paso 1”: usar SMOTE para balancear el conjunto de datos original (este paso se corresponde con el estado 1).

Figura 3.3: Esquema del funcionamiento de SMOTE-TCAD-2T



2. "Paso 2": construir el conjunto resultante (salida del algoritmo) incluyendo solo los ejemplos originales de la clase minoritaria.
3. "Paso 3": insertar en el *resultSet* todos aquellos ejemplos de la clase mayoritaria que tengan un grado pertenencia a la aproximación inferior de su clase por encima de un umbral dado como parámetro (δ_m).
4. "Paso 4":insertar en el *resultSet* todos aquellos ejemplos sintéticos que tengan un grado pertenencia a la aproximación inferior de su clase por encima de un umbral dado como parámetro (δ_s).
5. "Paso 5": Si los ejemplos encontrados en el paso 4 no alcanzan para balancear el conjunto y no se ha alcanzado un número máximo de iteraciones, entonces se comienza por el paso 1.

Este algoritmo ha sido aplicado para resolver un problema real de la ingeniería, en el siguiente capítulo se describe su utilización.

Algoritmo 3.2 Algoritmo propuesto SMOTE-TCAD-2T

Entrada: threshold for synthetic examples δ_s
threshold for majority class δ_m
iteration number n_i
SMOTE parameters
$neg[]$ array of negatives examples $pos[]$ array of positives examples
Salida: *resultSet*

1: Step1: Using SMOTE we create a set of synthetic data ($syntheticInstances[]$) for the minority class until the training set is balanced.
2: add $pos[]$ to *resultSet*
3: $executionNumber = 0$
4: $isbalance = false$
5: **while** ($executionNumber <= n_i$) & !($isbalance$) **do**
6: **for** $i \leftarrow 1$ **to** ($syntheticInstances[].lenght$) **do**
7: Compute $fuzzyLowerMembership(syntheticInstances[i])$
8: **if** $fuzzyLowerMembership[i] >= \delta_s$ **then**
9: insert $syntheticInstances[i]$ in *resultSet*
10: $nsynt++$
11: **end if**
12: **end for**
13: **for** $j \leftarrow 1$ **to** $neg[]$ **do**
14: Compute $fuzzyLowerMembership(neg[i])$
15: **if** $fuzzyLowerMembership[i] >= \delta_m$ **then**
16: insert $nneg[i]$ in *resultSet*
17: $nneg++$
18: **end if**
19: **end for**
20: $balance = (nneg = (nsynt + pos[].lenght))$
21: $executionNumber++$
22: **end while**
23: Step5: If there are no instances in the lower approximation, the solution is given as the set balanced with SMOTE, all synthetic instances are include in *resultSet*.

3.4. Aplicación del algoritmo SMOTE-TCAD-2T al diagnóstico de la necesidad de mantenimiento de los interruptores de alta potencia

En esta sección se estudia un problema real de la ingeniería haciendo uso de uno de los algoritmos propuestos en la tesis: el problema del diagnóstico de la necesidad de mantenimiento de los interruptores de alta potencia. Se aplica el algoritmo propuesto en el capítulo anterior SMOTE-TCAD-2T, el cuál es capaz de generar un nuevo conjunto de entrenamiento balanceado, donde las instancias sintéticas que han sido insertadas en la solución tienen alta calidad, y donde además han sido eliminadas las instancias de la clase mayoritaria con muy baja pertenencia a su clase. El estudio experimental realizado muestra cómo el algoritmo es capaz de mejorar considerablemente la predicción de la necesidad de mantenimiento, reduciendo en gran medida el número de falsos negativos.

3.4.1. Diagnóstico de la necesidad de mantenimiento

Actualmente el crecimiento acelerado del consumo de energía eléctrica tiene aparejado un incremento en las inversiones; desde la generación hasta la distribución, siendo necesaria su interconexión para asegurar un óptimo suministro de la energía con la calidad requerida Whitaker (1999). Todo ello conduce a realizar un cambio acelerado en la gestión del mantenimiento, principalmente en aspectos de tipo tecnológico, organizacional, documental y económico. Los interruptores de alta potencia (IAP en lo adelante), como uno de los elementos principales de una subestación eléctrica de cualquier tipo, (de transformación, transferencia o enlace), no escapan de ello.

El IAP es un dispositivo destinado al cierre y apertura de la continuidad de un circuito eléctrico, bajo carga y en condiciones normales, su comportamiento depende del nivel de confiabilidad de las redes eléctricas. El mismo debe ser capaz de interrumpir corrientes eléctricas de diferentes capacidades, pasando desde las corrientes capacitivas de varios cientos de amperes y las inductivas de varias decenas de kA, por ello asegurar su buen funcionamiento es de vital importancia para cualquier sistema eléctrico de potencia C37.06 (2000).

Para evitar rupturas que causan costosas fallas al sistema eléctrico los interruptores de potencia deben recibir mantenimiento cada cierto intervalo de tiempo, dichos intervalos son datos que brinda el fabricante del equipo. Los intervalos de tiempo indicados por los fabricantes se basan en valores experimentados durante varios años, sin embargo estos intervalos no suelen corresponderse con la realidad, pues regularmente los intervalos son más cortos ALSTOM (2002); debido a que muchas variables que deterioran su funcionamiento no son tenidas en cuenta, y esto a su vez provoca que la confiabilidad operacional de los interruptores disminuya e incluso puedan ocurrir averías imprevistas como se aborda en varios estudios Lindquist et al. (2008); Pitz and Weber (2001).

Actualmente importantes revistas científicas y congresos internacionales del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE por sus siglas en inglés); grupos de trabajos de asociaciones profesionales como el Consejo Internacional en Grandes Sistemas Eléctricos de Alta Tensión (CIGRE); el Instituto Americano de Normas Nacionales (ANSI por sus siglas en inglés), así como editoriales prestigiosas tales como Wiley & Sons, DEStech, Elseviers, Prentice-Hall y CRC Press se han dedicado a la divulgación y estudio de esta temática dada su importancia.

Desde el diseño de los primeros interruptores de potencia siempre se ha tenido en cuenta la confiabilidad y el mantenimiento de los mismos, varios de los estudios de confiabilidad siempre van asociados a análisis estadísticos, ejemplo de ello fueron los primeros estudios realizados a 77 892 interruptores de todos los tipos en 102 empresas de 22 países por el grupo de trabajo 13.06 de CIGRE en el período de 1974 a 1977 Mazza and Michaca (1981) mientras que en el segundo estudio se analizaron 70 708 interruptores de Hexafloruro de Azufre (SF6) Janssen (1994) en el cual se hizo un análisis profundo de la naturaleza de las fallas de mayor y menor importancia de acuerdo a la clasificación dada por (IEC 60694, 2002). También se han realizado otras investigaciones; en México se analizaron y estudiaron las fallas de una población de alrededor de 3 556 interruptores durante el período de 21 años iniciando en 1981 y concluyendo en el 2001; en Alemania durante un estudio de las redes eléctricas se incluyó también el análisis de la fatiga de varios interruptores de potencia de acuerdo a las corrientes de carga y cortocircuito; mientras que en Suecia y Finlandia se analizaron las fallas de alrededor de 1 546 interruptores Lindquist et al. (2008), en China el Instituto de Investigaciones de Potencia llevó a cabo un estudio entre

1999 y 2003 de varios interruptores de potencia mayores de 63kV y en Japón se analizó la confiabilidad y los esfuerzos de un numeroso grupo de interruptores.

Sin embargo, tras años de estudio aún no se cuenta con una metodología que logre predicciones certeras sobre la necesidad de mantenimiento del IAP, más allá de la dada por el fabricante. Cuando se desea saber si el IAP necesita mantenimiento, un equipo técnico debe abrirlo y realizar una revisión, éste será quien brinde el diagnóstico final. Abrir y manipular el interruptor es no es una tarea trivial, y en la mayoría de los casos el IAP no necesita mantenimiento (dado que la necesidad de mantenimiento es poco frecuente), también puede ocurrir que ya se demasiado tarde y haya que desechar el equipo.

3.4.2. El problema de los interruptores de alta potencia

Para todos los casos la necesidad de mantenimiento de un no es obvia dado el hecho de que éstos son usados, abiertos o cerrados, por un período extenso de tiempo. La necesidad de una predicción acertada se incrementa a lo largo del tiempo según la expansión del sistema de transmisión, dado que se transporta más energía en regiones más extensas. Es por ello que se usa un mantenimiento preventivo o, lo que es lo mismo, una planificación basada en tiempo fijo. Sin embargo, con el desarrollo de las tecnologías se han desarrollados nuevos enfoques para esta planificación C37.06 (2000). La planificación predictiva se basa en decidir realizar la inspección del equipo en intervalos de tiempo regulares. Este proceso incluye inspección objetiva (medida con las herramientas adecuadas) y subjetiva (medida por humanos) así como la reparación del problema (falla potencial). El objetivo es predecir la condición del interruptor con exactitud sin necesidad de abrirlo para inspección, aumentando su eficiencia y reduciendo considerablemente su costo. Este proceso se desarrolla normalmente por medio de pruebas, análisis estadístico o monitoreo de la condición.

La efectividad de la predicción de la necesidad de mantenimiento depende de la exactitud del análisis de la revisión visual, las pruebas y el análisis estadístico para determinar el nivel de daños del interruptor. Existen varias variables en la práctica que deben ser incluidas en dicho análisis. Estas variables son frecuentemente afectadas por el nivel de especialización

del experto. En esta investigación se usan técnicas de minería de datos para realizar dicha predicción y la tarea se divide en los siguientes pasos:

1. Construcción del conjunto de datos,
   a) Determinar las variables
   b) Realizar las mediciones para cada variable
   c) Etiquetar cada observación en: "necesita mantenimiento" o "no necesita mantenimiento" por un experto humano.
2. Determinar las características del conjunto de datos atendiendo al número de instancias por clases (balanceada o no).
3. Escoger el clasificador adecuado para la aplicación.
4. Escoger el algoritmo de preprocesamiento que mejores resultados ofrezca para ser usado en la aplicación.
5. Evaluar el comportamiento del sistema a través de un estudio experimental.

3.4.3. Construcción del conjunto de datos

Las variables usadas en nuestros datos fueron determinadas a partir de estudios internacionales, y de reglas y procedimientos propuestos por varios especialistas de compañías eléctricas. Se utilizó el algoritmo Delphi para consultar numerosos especialistas, con el fin de validar la relevancia de las variables que serán usadas para conformar el conjunto de entrenamiento. Estas consultas se aplicaron a un total de 35 especialistas altamente calificados del sector eléctrico, 25 de ellos nacionales y 10 extranjeros.

Para la creación del conjunto de entrenamiento, se escogieron aquellas variables que mostraron tener más influencia en la clase. De este procedimiento resultados 17 variables que demostraron tener un fuerte impacto en la decisión asociada a la necesidad de mantenimiento. Estas variables se muestran en la tabla 3.5.

Una vez decidido cuáles serían las variables se procedió a su medición. En este proceso los expertos dieron información relevante acerca del estado de los interruptores. A cada medición, los expertos le asignaron una etiqueta de clase (positiva/negativa). El conjunto

Tabla 3.5: Descripción de las variables usadas en el conjunto de datos IAP

Nombre de las variables	Tipo de variable
Aislamiento de Cámara	*real*
Aislamiento Apoyo	*real*
Aislamiento Total	*real*
Resistencia de Contacto	*real*
Presión SF6	*real*
Resistencia Bobinas	*real*
Cantidad Operaciones	*entero*
Terminales primarios	*nominal* $\{1-10\}$
Porcelanas	*nominal* $\{1-10\}$
Calefacción	*nominal* $\{1-10\}$
Estado de los Gabinetes	*nominal* $\{1-10\}$
Aterramiento	*nominal* $\{1-10\}$
Borneras	*nominal* $\{1-10\}$
Control General	*nominal* $\{1-10\}$
Criticidad Operacional	*nominal* $\{alto, medio, bajo\}$
Desgaste eléctrico	*nominal* $\{alto, medio, bajo\}$
Desgaste contactos	*nominal* $\{alto, medio, bajo\}$

de datos resultó tener un total de 369 muestras, 120 pertenecen a la clase positiva (necesidad de mantenimiento) mientras que 249 pertenecen a la clase negativa (no es necesario mantenimiento). En otras palabras, existen más de 2 ejemplos negativos por cada ejemplo positivo, por lo que estamos en presencia de un conjunto de datos no balanceado.

Como se explicó en el epígrafe 1.3.1 las técnicas para abordar el aprendizaje a partir de datos no balanceados están divididas en 3 grupos; en esta investigación nos centramos en las que están al nivel de los datos. Para abordar el problema del diagnóstico de la necesidad de mantenimiento de los IAP se propuso el algoritmo descrito en el capítulo anterior; el estudio experimental realizado se describe a continuación.

3.4.4. Estudio experimental

En este epígrafe se evaluará el comportamiento del algoritmo SMOTE-TCAD-2T sobre el conjunto de datos descrito en el epígrafe anterior. Se realizará el estudio comparativo con otros 9 algoritmos de preprocesamiento.

3.4.5. Configuración de los experimentos

El algoritmo de aprendizaje que se usó para el estudio es el árbol de decisión C4.5, este algoritmo que construye un árbol de decisión usando entropía de información, el mismo requiere de parámetros que han sido seleccionados atendiendo a la recomendación de su autor: particularmente, nivel de confianza = 0.25, número mínimo de itemsets por hojas = 2 y finalmente pruned = true.

Para comparar el comportamiento del algoritmo propuesto SMOTE-TCAD-2T frente al problema del diagnóstico de la necesidad de mantenimiento de los IAP, se seleccionaron 9 algoritmos de preprocesamiento, 7 de ellos descritos en 1.3.1 y los otros dos han sido propuestos en esta tesis: SMOTE-RSB$_*$ y SMOTE-TCAD. Los parámetros usados por SMOTE son los recomendados por su autor en Chawla et al. (2002).

La aplicación del algoritmo SMOTE-TCAD no resultó apropiada para resolver el problema de los IAP, debido a que en su versión original elimina todos los ejemplos de la clase mayoritaria. Se decide hacer una modificación a este método, la cual consiste en editar solo las instancias sintéticas usando dos valores de altos umbral ($\gamma = 0{,}8$ y $\gamma = 1$) esta variante la hemos llamado SMOTE-TCAD-S. El conjunto de datos fue particionado en 5 para realizar la validación cruzada (5 fold cross validation), la métrica usada para evaluar el comportamiento de los algoritmos es el AUC. Se utilizó la herramienta KEEL Alcalá et al. (2009, 2010) para el estudio experimental, todos los algoritmos usados han implementados allí.

3.4.6. Ajuste de parámetros de SMOTE-TCAD-2T

Como se mencionó en el epígrafe 3.3, el algoritmo SMOTE-TCAD-2T, necesita de 3 parámetros: el número de iteraciones T, el umbral γ_M para evaluar las instancias originales mayoritarias usando la región positiva (fuzzy-rough) POS y el umbral γ_S para evaluar los ejemplos sintéticos.

En Ramentol et al. (2012) se propuso el uso de 10 iteraciones, en este estudio se utilizará 5, $\gamma_S = 1$ para evaluar los ejemplos sintéticos y $\gamma_M = 0{,}03$ para editar los ejemplos mayoritarios.

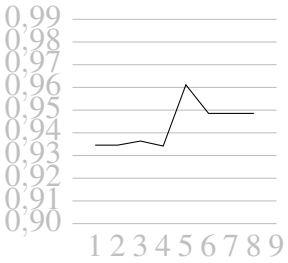
Estos parámetros fueron fijados con en esos valores siguiendo las siguientes premisas:

- El número de iteraciones fue limitado a 5, porque dada la necesidad de ejemplos que hace falta generar y la calidad que se está exigiendo a dichos ejemplos, 5 iteraciones deben ser suficientes para que haya convergencia.
- El grado de pertenencia de una instancia a la región positiva está entre 0 y 1. Una instancia que pertenezca con grado 1 a dicha región es sin ninguna duda una buena representación de su clase. Teniendo en cuenta que estamos trabajando con una aplicación real, donde cada nuevo ejemplo sintético representa un estado en el que un IAP necesita mantenimiento, y teniendo en cuenta lo que esto significa para un sistema eléctrico, se debe ser sumamente cuidadoso para introducir un ejemplo sintético en los datos, ya que será un ejemplo que no ha sido observado en la realidad. Esta es la razón por la cual solo se introducen los ejemplos que son una buena representación de su clase, y por ello el umbral ha sido fijado en el máximo valor que este puede alcanzar (1).
- Para el caso de γ_M , se sigue una idea semejante a la anterior. Cada ejemplo original presente en el conjunto de datos representa una observación real de los ingenieros sobre el IAP, por tanto se debe ser muy cuidadoso a la hora de eliminar cualquiera de estos ejemplos porque se puede estar perdiendo conocimiento muy valioso. Un grado de pertenencia muy cercano a cero, representa un ejemplo que es muy cercano a la clase contraria, un ejemplo que afecta considerablemente la separabilidad de las clases durante el proceso de clasificación, que pudiera ser incluso ruido. En esta investigación se decide eliminar todos aquellos ejemplos negativos originales que tengan un grado de pertenencia a la región positiva por debajo de 0.03. Este algoritmo de limpieza no es aplicable a los ejemplos minoritarios originales debido a que cada uno de ellos es un ejemplo "raro" que ocurre con baja frecuencia.

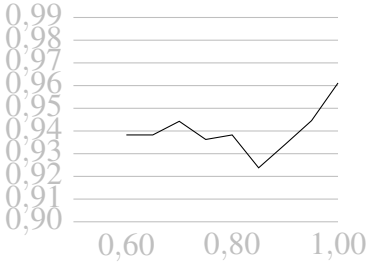
La figura 3.4 ilustra el procedimiento seguido para el ajuste de los parámetros. La figura 3.4a muestra por el eje Y el AUC obtenido usando C4.5 en cada iteración representada en

Figura 3.4: **Parameters** – Ajustando los parámetros finales, manteniendo el resto constantes

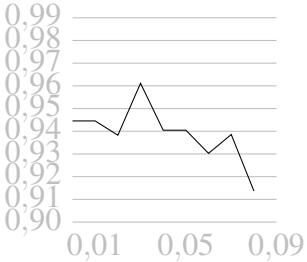
(a) **Iteraciones** - Ajustando el número de iteraciones



(b) **Umbral 1** - Ajustando el umbral para las instancias sintéticas



(c) **Umbral 2** - Ajustando el umbral para las instancias mayoritarias



el eje X y manteniendo el resto de los parámetros constantes en los valores indicados en los experimentos. La figura 3.4b muestra el mismo resultado por el eje Y pero asociado a la γ_S por eje X. La figura J.2c representa cómo el AUC (representado en el eje Y) varía mientras es variado el parámetro γ_M (asociado al eje X) y manteniendo constantes el resto de los parámetros con los valores indicados en los experimentos.

Como se puede observar durante el ajuste del parámetro γ_S , el mejor AUC es obtenido cuando el umbral es 1, lo cual permite la inserción de los mejores ejemplos sintéticos en el conjunto final. El efecto contrario ocurre con el γ_M , mientras mayor es el valor menor es el AUC obtenido, lo que indica que subir este umbral implica la pérdida de instancias importantes en la etapa de clasificación, un valor muy cercano a cero nos lleva solo a eliminar algunos ejemplos que pudieran considerarse ruidosos.

3.4.7. Comparación con algoritmos del estado del arte

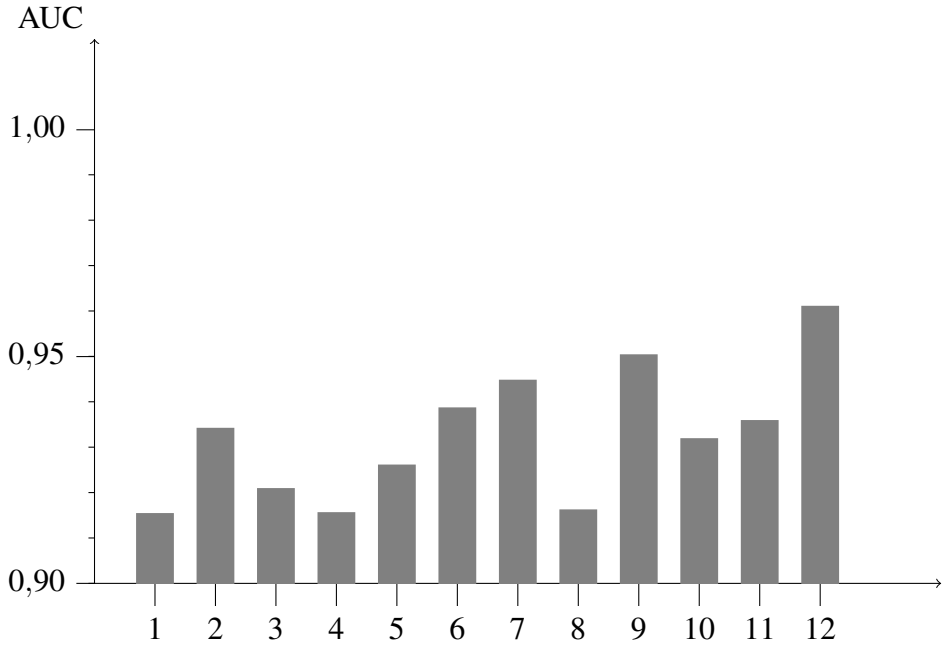
Los resultados del estudio experimental realizado se muestran en la tabla 3.6, en la columna (tra) se muestran los resultados sobre el conjunto de entrenamiento y en la columna (tst) los resultados sobre el conjunto de prueba. De igual forma en la tabla 3.5 se muestran gráficamente los resultados sobre el conjunto de prueba, en el eje Y se muestra el AUC mientras que en el eje X se muestran los algoritmos de preprocesamiento, cada número es el asociado a los algoritmos en la tabla 2. Para enfatizar en las ventajas del uso de preprocesamiento en los conjuntos no balanceados, en la primera fila se muestra los resultados que alcanza el C4.5 sobre el conjunto original.

Como se puede observar en la gráfica, la mayoría de los algoritmos de preprocesamiento mejoran el resultado del C4.5 sin preprocesamiento. El algoritmo que mejores resultados alcanza es el SMOTE-TCAD-2T, seguido por el SMOTE-RSB* por solo 1 porciento.

Tabla 3.6: Comparación del AUC para entrenamiento (tra) y prueba (tst)

	Resampling method	tra	tst
1	Original	0.9588	0.9155
2	SMOTE	0.9830	0.9343
3	SMOTE-TL	0.9601	0.9210
4	SMOTE-ENN	0.9568	0.9157
5	Borderline-SMOTE1	0.9666	0.9262
6	Borderline-SMOTE2	0.9782	0.9388
7	Safe-Level-SMOTE	0.9568	0.9449
8	SPIDER2	0.9509	0.9163
9	SMOTE-RSB*	0.9839	0.9505
10	SMOTE-TCAD-S($\gamma = 0{,}8$)	0.9834	0.9320
11	SMOTE-TCAD-S($\gamma = 1$)	0.9761	0.9360
12	SMOTE-TCAD-2T	0.9798	0.9612

Figura 3.5: AUC para cada algoritmo en prueba



Nos gustaría además enfatizar en la comparación de SMOTE-TCAD-2T con la introducida en Ramentol et al. (2012), descrita en el epígrafe 3.1. Como se puede observar en la tabla 3.6, SMOTE-TCAD-S obtiene el mayor resultado en entrenamiento pero falla considerablemente en el conjunto de prueba para ambos de valores de γ. En la tabla 3.7 se muestra para cada partición la cantidad de instancias que quedan por clase después de ejecutados los algoritmos SMOTE-TCAD-S $(\gamma = 0{,}8)$ y SMOTE-TCAD-2T. Cada fila etiquetada como "#may. eliminados" muestra el número de ejemplos mayoritarios que fueron eliminados. Cada fila etiquetada como "#sint. creados" muestra el número de ejemplos sintéticos creados.

Tabla 3.7: Instancias creadas y eliminadas por SMOTE-TCAD-S y SMOTE-TCAD-2T

		SMOTE-TCAD-S	SMOTE-TCAD-2T
Partición 1	**#may eliminados**	0	5
	#synt creados	103	98
Partición 2	**#may eliminados**	0	6
	#synt creados	103	99
Partición 3	**#may eliminados**	0	7
	#synt creados	103	99
Partición 4	**#may eliminados**	0	7
	#synt creados	103	96
Partición 5	**#may eliminados**	0	7
	#synt creados	104	97

Como se ha mencionado, SMOTE-TCAD-S no elimina ningún ejemplo de la clase mayoritaria, sin embargo el algoritmo SMOTE-TCAD-2T ha demostrado que usar un valor muy pequeño para editar la clase mayoritaria resulta beneficioso, pues solo se eliminan muy pocas instancias por partición y son aquellas que tienen grandes probabilidades de ser ruido. Por otra parte, el uso de un valor alto para seleccionar las sintéticas que estarán en el conjunto final, nos da la posibilidad de obtener ejemplos de alta calidad que ayudan a mejorar el comportamiento de los clasificadores frente a estos conjuntos.

Desde el punto de vista de los ingenieros eléctricos, el objetivo fundamental es disminuir el número de falsos negativos, es decir, disminuir el número de errores al decir que un IAP necesita mantenimiento cuando no es así. Si el IAP no necesita mantenimiento y el sistema dice “sí”, el costo asociado será el de tener que hacer pruebas al equipo y tal vez dar mantenimiento en algunas partes necesarias. Sin embargo si el sistema predice “no” cuando el equipo realmente está necesitando mantenimiento, el daño es mucho peor, y puede terminar ocurriendo una falla importante en el sistema eléctrico. Otro daño pudiera ser la pérdida de sincronismo del generador, o la falla total y ruptura del interruptor. El costo económico asociado pudiera ser el reemplazo del equipo (que es costoso) sin mencionar las pérdidas asociadas a las fallas eléctricas que pueden producirse por fallos en el funcionamiento del IAP.

Por esta razón, se investigó a fondo en la matriz de confusión, la cual se muestra en la tabla 3.8 para todos los algoritmos que se aplicaron. La tabla muestra como nuestra propuesta es la que menos falla en la clasificación de ejemplos positivos. El algoritmo SMOTE-TL también falla en solo 5 ejemplos positivos al igual que nuestro método, sin embargo produce 29 fallos en la clase negativa, 20 más que nuestro método.

Tabla 3.8: Matriz de confusión para los algoritmos comparados

		Predicted negative	Predicted positive
Original	Real negative	238	11
	Real positive	15	105
SMOTE	Real negative	237	12
	Real positive	10	110
SMOTE-TL	Real negative	220	29
	Real positive	5	115
SMOTE-ENN	Real negative	236	13
	Real positive	14	106
Borderline-SMOTE1	Real negative	231	18
	Real positive	9	111
Borderline-SMOTE2	Real negative	231	18
	Real positive	6	114
Safe-Level-SMOTE	Real negative	234	15
	Real negative	6	114
SPIDER2	Real negative	226	23
	Real positive	9	111
SMOTE-RSB*	Real negative	243	6
	Real positive	9	111
SMOTE-TCAD-S($\gamma = 0{,}8$)	Real negative	242	7
	Real positive	12	108
SMOTE-TCAD-S($\gamma = 1$)	Real negative	240	9
	Real positive	12	108
SMOTE-TCAD-2T	Real negative	240	9
	Real positive	5	115

3.5. Conclusiones parciales

En este capítulo fueron presentadas dos propuestas basadas en el enfoque difuso de la TCA. Ambas propuestas utilizan el concepto de región positiva para determinar la calidad de los ejemplos sintéticos y los originales mayoritarios.

La primera propuesta SMOTE-TCAD, logra evaluar las instancias sintéticas que genera SMOTE y las originales mayoritarias, dejando intactas las originales minoritarias; logrando de esta forma obtener instancias sintéticas que pertenezcan en un alto grado a la región positiva de su clase, y dejar solo una buena representación de la clase mayoritaria, usando el mismo criterio de selección.

La segunda propuesta SMOTE-TCAD-2T, edita las instancias sintéticas y las mayoritarias originales, pero usando umbrales diferentes para ambas ediciones. Para las sintéticas se usa un umbral muy elevado que permite exigir la total pertenencia de las instancias a la región positiva de su clase; para las mayoritarias se usa un valor muy pequeño que permite la sola eliminación de los ejemplos que pudieran resultar ruidosos en la fase de clasificación.

La principal novedad de estas propuestas es que el uso del enfoque difuso permite "suavizar" la pertenencia de las instancias a la región positiva, y obtener de forma gradual esta pertenencia, con este enfoque es posible editar también la clase mayoritaria sin el riesgo de eliminar demasiados ejemplos.

La utilización del algoritmo SMOTE-TCAD-2T como un paso de preprocesamiento intermedio para la predicción de la necesidad de mantenimiento de los interruptores de alta potencia. Las principales contribuciones desde el punto de vista del aprendizaje automático se pueden resumir en:

- El uso de 2 umbrales permite la edición de manera diferente de los ejemplos negativos y sintéticos.
- La no eliminación de ejemplos positivos originales, debido a que cada uno de ellos representa un suceso no común, es decir, la ocurrencia real de una situación en la que el IAP requiere mantenimiento, por lo que cada instancia positiva constituye un ejemplo muy valioso para la etapa de aprendizaje.
- La eliminación de aquellas instancias mayoritarias con muy baja pertenencia a la región positiva de su clase.
- Solamente los ejemplos sintéticos con el valor más de alto de pertenencia a la región positiva se insertan en el conjunto final.
- El algoritmo SMOTE-TCAD-2T obtiene mejores resultados que 9 algoritmos reconocidos del estado del arte.

Resumiendo, el algoritmo es capaz de reducir de manera significativa el número de ejemplos mal clasificados. Desde el punto de vista de los ingenieros eléctricos, esto se traduce en evitar a tiempo fallas y rupturas del IAP, necesidad del reemplazo del IAP, evitar fallas en el sistema eléctrico, aperturas innecesarias del equipo cuando este no requiere mantenimiento, en fin un ahorro considerable de recursos y de fallas eléctricas.

CAPÍTULO 4

Un nuevo algoritmo para clasificación desbalanceada usando la teoría de los conjuntos aproximados difusa y la agregación con el operador OWA utilizando vectores de pesos

"The ability to focus attention on important things is a defining characteristic of intelligence."
– Robert Shiller.

En este capítulo se propone un nuevo algoritmo de clasificación para conjuntos de datos no balanceados. La propuesta es una extensión para dominios no balanceados del algoritmo de los Vecinos más Cercanos con Conjuntos Aproximados Difusos, VCCAD, (Fuzzy Rough Nearest Neighbor FRNN) Jensen and Cornelis (2011).

El algoritmo FRNN tiene una importante debilidad: su manera de clasificar está determinada completamente por los ejemplos más cercados en cada clase, esto lo hace muy sensible al ruido. Por otra parte FRNN trata la clase positiva y la negativa en una forma

simétrica, por lo que no está preparado para lidiar con el desbalance de clases. En esta investigación se propone calcular $\underline{P}(x)$ y $\underline{N}(x)$ teniendo en cuenta no solo los ejemplos más cercanos de la clase contraria, sino *todos* los ejemplos de la clase contraria, asignándoles a éstos pesos decrecientes proporcionales a su similaridad con el ejemplo de prueba x. Análogamente también consideramos una forma más flexible de calcular la aproximación superior para $\overline{P}(x)$ y $\overline{N}(x)$. Esto se logra a través del uso de vectores de peso ordenados (ordered weighted average OWA) del modelo híbrido difuso-aproximado presentado en Cornelis et al. (2010).

En las siguientes secciones se explicará en detalles el algoritmo base de nuestra propuesta FRNN, los pasos detallados de nuestro algoritmo, las diferentes estrategias de ponderación para la construcción de agregación con el operador OWA, se describirá cómo se calcula la métrica AUC que utilizaremos para evaluar el desempeño de nuestra propuesta, se mostrará el estudio experimental realizado así como el análisis estadístico y las conclusiones.

4.1. Algoritmo Vecinos más Cercanos con Conjuntos Aproximados Difusos (FRNN)

En esta sección estudiaremos el algoritmo de clasificación Vecinos más Cercanos con Conjuntos Aproximados Difusos (FRNN por sus siglas en inglés) propuesto en Jensen and Cornelis (2011). Dicho algoritmo lo hemos aplicado directamente a un caso específico de un problema no balanceado de dos clases. En el dominio de los no balanceados, formalizamos el problema como:

Consideramos un conjunto de datos $U = U_{tr} \cup U_{ts}$, consistente en un conjunto de entrenamiento U_{tr} y un conjunto de prueba U_{ts}. Todas las instancias estas descritas por un conjunto de características $\mathscr{A} = \{a_1, \ldots, a_m\}$, sus atributos. Además, el conjunto de entrenamiento $U_{tr} = P \cup N$, donde P representa la clase positiva y N la clase negativa. Denotamos $p = |P|$, $n = |N|$ y $t = p + n$.

Para predecir la clase de una nueva instancia del conjunto de prueba x, el algoritmo VCCAD calcula la suma de los grados de pertenencia de x a la aproximación inferior y superior de cada clase, y asigna la instancia a la clase cuya suma haya sido mayor. Más específicamente,

dado un implicator [1] $\mathscr{I}$, una t-norm $\mathscr{T}$, y una relación difusa R en U que representa una inseparabilidad aproximada, el grado de pertenencia de x a la aproximación inferior de P y N bajo R se define por, respectivamente,

$$\underline{P}(x) = \min_{y \in U_{tr}} \mathscr{I}(R(x,y), P(y)) \tag{4.1}$$

$$\underline{N}(x) = \min_{y \in U_{tr}} \mathscr{I}(R(x,y), N(y)), \tag{4.2}$$

Mientras que los grados de pertenencia de x a la aproximación superior de P y N bajo R se define por, respectivamente,

$$\overline{P}(x) = \max_{y \in U_{tr}} \mathscr{T}(R(x,y), P(y)) \tag{4.3}$$

$$\overline{N}(x) = \max_{y \in U_{tr}} \mathscr{T}(R(x,y), N(y)). \tag{4.4}$$

En esta investigación consideramos $\mathscr{I}$ y $\mathscr{T}$ definidos por $\mathscr{I}(a,b) = \max(1-a,b)$ y $\mathscr{T}(a,b) = \min(a,b)$, para a,b en $[0,1]$. Se puede verificar que para este caso, las ecuaciones (4.1)–(4.4) pueden ser simplificadas a:

$$\underline{P}(x) = \min_{y \in N} 1 - R(x,y) \tag{4.5}$$

$$\underline{N}(x) = \min_{y \in P} 1 - R(x,y) \tag{4.6}$$

$$\overline{P}(x) = \max_{y \in P} R(x,y) \tag{4.7}$$

$$\overline{N}(x) = \max_{y \in N} R(x,y) \tag{4.8}$$

[1] Un implicator $\mathscr{I}$ es una asociación $[0,1]^2 \to [0,1]$ que disminuye en su primer argumento y aumenta en su segundo y que satisface $\mathscr{I}(0,0) = \mathscr{I}(0,1) = \mathscr{I}(1,1) = 1$ y $\mathscr{I}(1,0) = 0$.

En otras palabras, $\underline{P}(x)$ es determinado por la distancia a su ejemplo negativo más cercano (mayoritario), y $\underline{N}(x)$ es determinado por la distancia a su ejemplo positivo más cercano (minoritario). Por otra parte para determinar $\overline{P}(x)$ y $\overline{N}(x)$, buscamos el ejemplo más similar a x perteneciente a la clase positiva o negativa respectivamente. Obviamente las aproximaciones inferior y superior son complementarias: $\overline{P}(x) = 1 - \underline{N}(x)$ and $\overline{N}(x) = 1 - \underline{P}(x)$. El algoritmo VCCAD determina la clasificación de una instancia de prueba x en U de la siguiente forma. Calculamos:

$$\mu_P(x) = \frac{\underline{P}(x) + \overline{P}(x)}{2} = \frac{\underline{P}(x) + 1 - \underline{N}(x)}{2} \tag{4.9}$$

$$\mu_N(x) = \frac{\underline{N}(x) + \overline{N}(x)}{2} = \frac{\underline{N}(x) + 1 - \underline{P}(x)}{2} \tag{4.10}$$

$x$es clasificado como ejemplo positivo si $\mu_P(x) \geq \mu_N(x)$, en otro caso es clasificado como ejemplo negativo.

4.2. Algoritmo propuesto: Imbalanced Fuzzy-Rough Ordered Weighted Average Nearest Neighbor (IFROWANN)

En la sección 4.2.1 se introduce nuestro modelo de clasificación difuso-aproximado con uso del operador OWA (Imbalanced Fuzzy-Rough Ordered Weighted Average Nearest Neighbor IFROWANN) para conjuntos no balanceados, mientras que en la sección 4.2.2 proponemos 6 estrategias para construir los operadores OWA para las clases positiva y negativa.

4.2.1. Descripción del algoritmo

Primeramente recordaremos en qué consiste un operador OWA. Dada un secuencia A de t valores reales $A = \langle a_1, \ldots, a_t \rangle$, y un vector de pesos $W = \langle w_1, \ldots, w_t \rangle$ tal que $w_i \in [0,1]$ y $\sum_{i=1}^{t} w_i = 1$ la agregación OWA de A por W está dada por:

$$OWA_W(A) = \sum_{i=1}^{t} w_i b_i$$

donde $b_i = a_j$ si a_j es el elemento i^{th} más grande en A. Por ejemplo si $A = \langle 3,1,2 \rangle$, y $W = \langle 0{,}3, 0{,}2, 0{,}5 \rangle$, entonces

$$OWA_W(A) = 0{,}3 * 3 + 0{,}2 * 2 + 0{,}5 * 1 = 1{,}8$$

Como casos especiales del operador OWA se tienen los operadores mínimo y máximo. Nótese que, si $W = \langle 0{,}0 \ldots, 1 \rangle$, entonces $OWA_W(A)$ retornará el valor mínimo en A, mientras que $W = \langle 1, 0 \ldots, \rangle$ provocará que $OWA_W(A)$ sea máximo en A. Sin embargo, podemos considerar los vectores de pesos OWA para modelar una gran variedad de estrategias de agregación distintas desde el min hasta el max en las ecuaciones (4.1)–(4.4). En general, dado los vectores de pesos OWA W_P^l y W_N^l, podemos definir la W_P^l-aproximación inferior de P bajo R, y la W_N^l-aproximación inferior de P bajo R como:

$$\underline{P}_{W_P^l}(x) = \underset{y \in U_{tr}}{OWA_{W_P^l}} \langle \mathscr{I}(R(x,y), P(y)) \rangle \tag{4.11}$$

$$\underline{N}_{W_N^l}(x) = \underset{y \in U_{tr}}{OWA_{W_N^l}} \langle \mathscr{I}(R(x,y), N(y)) \rangle, \tag{4.12}$$

donde R es una relación de tolerancia difusa y $\mathscr{I}$ es el implicator. Por otra parte, dados los vectores de pesos OWA W_P^u y W_N^u y una t-norm $\mathscr{T}$, podemos definir la W_P^u-aproximación superior de P bajo R, y la W_N^u-aproximación superior de P bajo R como:

$$\overline{P}_{W_P^u}(x) = \underset{y \in U_{tr}}{OWA_{W_P^u}} \langle \mathscr{T}(R(x,y), P(y)) \rangle \tag{4.13}$$

$$\overline{N}_{W_N^u}(x) = \underset{y \in U_{tr}}{OWA_{W_N^u}} \langle \mathscr{T}(R(x,y), N(y)) \rangle, \tag{4.14}$$

Con el objetivo de mantener el complemento entre las dos aproximaciones, hemos impuesto la condición $(W_P^u)_i = (W_N^l)_{t-i+1}$ y $(W_N^u)_i = (W_P^l)_{t-i+1}$, para $i = 1, \ldots, t$.

Puede verificarse que bajo esta restricción se cumple que $\overline{P}_{W_P^l}(x) = 1 - \underline{N}_{W_N^u}(x)$ y $\overline{N}_{W_N^l}(x) = 1 - \underline{P}_{W_P^u}(x)$.

El algoritmo FROWA entonces determina la clasificación para una instancia x en U del conjunto de prueba de la siguiente forma. Calculamos

$$\begin{aligned} \mu_P(x) &= \frac{\underline{P}_{W_P^l}(x) + \overline{P}_{W_P^u}(x)}{2} \\ &= \frac{\underline{P}_{W_P^l}(x) + 1 - \underline{N}_{W_N^l}(x)}{2} \end{aligned} \tag{4.15}$$

$$\begin{aligned} \mu_N(x) &= \frac{\underline{N}_{W_N^l}(x) + \overline{N}_{W_N^u}(x)}{2} \\ &= \frac{\underline{N}_{W_N^l}(x) + 1 - \underline{P}_{W_P^l}(x)}{2} \end{aligned} \tag{4.16}$$

Una vez más x es clasificada como positiva si $\mu_P(x) \geq \mu_N(x)$, en otro caso es clasificada como negativa.

4.2.2. La agregación con el operador OWA usando vectores de ponderación para la clasificación de datos no balanceados

Un factor de suma importancia en la aplicación de nuestro algoritmo IFROWANN es la selección del vector de peso OWA en las ecuaciones (4.11)–(4.14). Dado que asumimos la complementariedad entre la aproximación inferior y superior, en esta sección nos referiremos únicamente a la inferior. Particularmente, hemos diseñado vectores que brindan generalizaciones flexibles del operador mínimo, y a la vez tienen en cuenta el desbalance presente en los datos. Primero, nótese que: bajo el supuesto $\mathscr{I}(R(x,y),P(y)) = 1$ si $P(y) = 1$, en otras palabras, cuando el ejemplo y es positivo. De forma similar $\mathscr{I}(R(x,y),N(y)) = 1$ cuando y es negativo. Pudiera decirse que estos valores no deben ser tomados en cuenta cuando se calcula la aproximación inferior; sin lugar a dudas, de acuerdo con la interpretación de la teoría de los conjuntos aproximados, una instancia pertenece a la aproximación inferior de una clase si la misma puede ser "separada" de las instancias que pertenecen a clases diferentes. En el contexto del algoritmo FROWA propuesto, podemos

implementar esta idea asignando peso cero a las correspondientes posiciones del vector OWA. En particular, las primeras p posiciones en W_P^l se les puede asignar valor cero, teniendo en cuenta que estas se corresponden con los valores más altos de $\mathscr{I}(R(x,y),P(y))$, y también a las p instancias positivas en el conjunto de entrenamiento. De igual forma, las primeras n posiciones en W_N^l se les asigna peso cero. El resto de las posiciones en el vector W_P^l corresponde a las instancias en N. Para estas instancias, el valor de la implicación es igual a $\mathscr{I}(R(x,y),P(y)) = \text{máx}(1-R(x,y),0) = 1-R(x,y)$. Consideramos dos estrategias alternativas de construcción de pesos, ambos asignan pesos altos a los valores de implicación pequeños.

$$\begin{aligned} W_P^{l_1} &= \left\langle 0.\ldots,0.\frac{2}{n(n+1)},\frac{4}{n(n+1)},\ldots,\right. \\ &\quad \left.\frac{2(n-1)}{n(n+1)},\frac{2}{n+1}\right\rangle \end{aligned} \tag{4.17}$$

$$\begin{aligned} W_P^{l_2} &= \left\langle 0.\ldots,0.\frac{1}{2^n-1},\frac{2}{2^n-1},\ldots,\right. \\ &\quad \left.\frac{2^{n-2}}{2^n-1},\frac{2^{n-1}}{2^n-1}\right\rangle \end{aligned} \tag{4.18}$$

La principal diferencia entre ambos vectores es que en el segundo caso los pesos decrecen más rápidamente que en el primero. Por ejemplo, si $n = 5$, entonces:

$$W_P^{l_1} = \left\langle 0.\ldots,0.\frac{1}{15},\frac{2}{15},\frac{3}{15},\frac{4}{15},\frac{5}{15}\right\rangle \tag{4.19}$$

$$W_P^{l_2} = \left\langle 0.\ldots,0.\frac{1}{31},\frac{2}{31},\frac{4}{31},\frac{8}{31},\frac{16}{31}\right\rangle \tag{4.20}$$

De una forma completamente análoga, podemos obtener dos versiones de los vectores de pesos $W_N^{l_1}$.

$$\begin{aligned} W_N^{l_1} &= \left\langle 0\ldots,0.\frac{2}{p(p+1)},\frac{4}{p(p+1)},\ldots,\right. \\ &\quad \left.\frac{2(p-1)}{p(p+1)},\frac{2}{p+1}\right\rangle \end{aligned} \tag{4.21}$$

$$\begin{aligned} W_N^{l_2} &= \left\langle 0\ldots,0.\frac{1}{2^p-1},\frac{2}{2^p-1},\ldots,\right. \\ &\quad \left.\frac{2^{p-2}}{2^p-1},\frac{2^{p-1}}{2^p-1}\right\rangle \end{aligned} \tag{4.22}$$

Partiendo del hecho de que generalmente p es mucho más pequeño que n, podemos esperar que el vector de pesos obtenidos para las clases positiva y negativa sean bastante diferentes. Sin embargo, en la práctica, incluso para valores relativamente pequeños de n y p los vectores $W_P^{l_2}$ y $W_N^{l_2}$ rápidamente tienden a ser fijos,

$$W = \left\langle \ldots,\frac{1}{32},\frac{1}{16},\frac{1}{8},\frac{1}{4},\frac{1}{2}\right\rangle, \tag{4.23}$$

de esta forma el desbalance de clases no es tomado en cuenta. Por esta razón, en nuestros experimentos consideramos dos variantes mixtas, donde se usan por ejemplo $W_P^{l_1}$ y $W_N^{l_2}$. Por otra parte cuando n es muy grande, todos los pesos en $W_P^{l_1}$ se vuelven muy pequeños, un fenómeno similar ocurre en el clasificador kNN cuando los k es muy alto. Con el objetivo de mitigar este efecto, consideramos la siguiente variante para el vector $W_P^{l_1}$, dado $0 \leq \gamma \leq 1$,

$$\begin{aligned} W_P^{l_1,\gamma} &= \left\langle 0\ldots,0.\frac{2}{r(r+1)},\frac{4}{r(r+1)},\ldots,\right. \\ &\quad \left.\frac{2(r-1)}{r(r+1)},\frac{2}{r+1}\right\rangle \end{aligned} \tag{4.24}$$

donde $r = \lceil p+\gamma(n-p)\rceil$. Claramente, $W_P^{l_1,0} = W_N^{l_1}$ y $W_P^{l_1,1} = W_P^{l}$.

4.3. Configuración de experimentos

En esta sección se presentan los parámetros y configuraciones necesarios para realizar el estudio experimental. Se describen los conjuntos de datos que se utilizarán, la métrica de evaluación, así como los parámetros de los algoritmos propuestos y los del estado del arte. Además se realiza un análisis para determinar el mejor valor de k para el clasificador kNN, para ello se prueban 100 valores distintos que van en puntos equidistantes desde 1 hasta el total de ejemplos en el conjunto de entrenamiento.

4.3.1. Métrica de evaluación

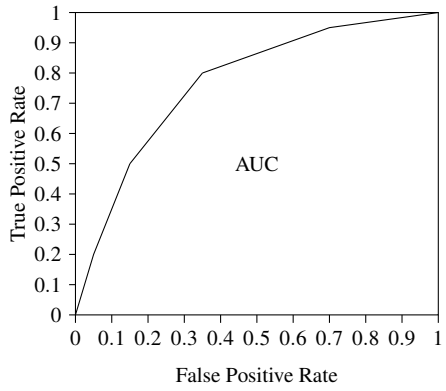
Como se explicó en la sección 1.3.6, las métricas de evaluación tradicionales para clasificadores no resultan apropiadas en el contexto de los no balanceados. En uso del Área bajo la curva ROC (AUC), ha demostrado ser una métrica que no es sensible al desbalance por clases y que evalúa de forma "justa" el comportamiento del clasificador frente a conjuntos con desigual representación en sus clases.

Para explicar las curvas ROC es necesario en primer lugar recordar en qué consiste un clasificador probabilístico. Los clasificadores probabilísticos calculan para cada instancia la probabilidad de pertenencia a la clase positiva p_+ y la probabilidad de pertenencia a la clase negativa p_-. Para tomar una decisión final en la predicción de la clase se fija un umbral θ. Cuando un $p_+ \geq \theta$, por ejemplo, la instancia será clasificada como positiva con probabilidad mayor que θ. De forma para cada valor posible del umbral θ se obtendrá un clasificador tradicional o discreto.

Un algoritmo basado en umbral permitirá que una instancia sea clasificada como positiva aun cuando $p_- > p_+$, además de que valores muy pequeños de θ pudieran ocasionar que muchas instancias negativas fueran clasificadas como positivas, mientras que valores altos de θ garantizan que las instancias positivas sean clasificadas como positivas. Obviamente la relación entre los verdaderos positivos y los falsos positivos depende del valor del umbral fijado, y la robustez de un clasificador será evaluada por la forma en la maneja con esta situación.

Una curva ROC es construida en un espacio bidimensional, graficando el índice de falsos positivos (FPR) y en índice de verdaderos positivos en los ejes X e Y respectivamente. La figura 4.1 muestra una representación visual de la curva ROC.

Figure 4.1: El Área bajo la curva ROC



Dada la matriz de confusión de un clasificador discreto C, se pueden obtener FPR_C y FPR_C. Como resultado, un clasificador discreto genera el punto (FPR_C, FPR_C) en el espacio ROC, el punto $(0,1)$ corresponde a una clasificación perfecta, es decir, la separación de las clases en espacios diferentes.

Para el caso de un clasificador probabilístico, para cada valor del umbral θ un clasificador discreto genera un punto del espacio ROC, variando θ es posible obtener una curva continua. La curva $x = y$ corresponde a una clasificación aleatoria, es decir, para cada valor de θ fijado TPR y FPR son iguales, lo cual significa que los elementos de las 2 clases son igualmente clasificados como positivos. Las curvas más empinadas corresponden a los clasificadores que mejor logran discriminar entre ambas clases.

Como ya hemos dicho anteriormente, las curvas ROC son insensibles a los cambios en la distribución por clases, es por ello que resulta una métrica muy apropiada cuando se trabaja con datos no balanceados. Como en Fawcett (2006), esta conclusión parte del hecho de que las curvas ROC son calculadas usando índices de filas de la matriz de confusión. El índice de ejemplos positivos a negativos corresponde al índice de las sumas de la fila. Como las filas son usadas por separado, la curva ROC no depende de la distribución actual por clases.

La información contenida en la curva ROC puede ser representada como un escalar, el *Área bajo la Curva ROC* (AUC), la cual corresponde al área contenida entre la curva y el eje horizontal en el intervalo $[0, 1]$. El AUC puede ser interpretado como la probabilidad de que el clasificador asigne a una instancia negativa escogida aleatoriamente, un valor más pequeño de probabilidad de pertenencia a la clase positiva que el asignado a una instancia positiva escogida aleatoriamente Fawcett (2006).

En nuestros experimentos seguimos lo planteado en Fawcett (2006) y calculamos el AUC a través de la aproximación de la curva continua ROC por un número finito de puntos. Las coordenadas de estos puntos en el espacio ROC son los índices de falsos positivos y verdaderos positivos obtenidos variando el parámetro θ de la probabilidad que tiene cada instancia de clasificada como positiva. La curva en sí es aproximada por la interpolación lineal entre los puntos calculados. El AUC puede determinarse entonces como la suma de las áreas de sucesivas de los trapezoides. Este algoritmo es conocido como la regla del trapezoide (trapezoid rule) y se detalla en el algoritmo 4.1.

4.3.2. Conjuntos de datos utilizados

Para evaluar nuestra propuesta se consideraron 102 conjuntos de datos Asuncion and Newman (2007) de diferentes índices de desbalance (entre 1.82 y 129.44). Los conjuntos de datos fueron obtenidos a partir de modificar conjuntos con múltiples clases en problemas no balanceados de dos clases. La descripción de los conjuntos se muestra en la tabla 4.1. La columna indica el índice de desbalance Orriols-Puig and Bernado-Mansilla (2009) (imbalanced ratio), que se define como el cociente entre la cantidad de ejemplos en la clase mayoritaria y la cantidad en la clase minoritaria; la columna Inst. indica el número de instancias y Attr. el número de atributos.

Cada conjunto ha sido particionado con el objetivo de realizar una validación cruzada usando 5 particiones (fivefolds cross-validation), el 80 % es usado como conjunto de entrenamiento y el 20 % como conjunto de prueba, donde las 5 particiones de prueba conforman el conjunto completo. Para cada conjunto el resultado considerado es el promedio de las 5 particiones. Las particiones fueron realizadas de forma tal que el número

Algoritmo 4.1 Calculando el área bajo la curva ROC

Entrada: L, For each instance $x_i \in U$ $(i = 1,\ldots,t)$, the estimated probability p^i_+ and its true class $l(x_i)$

Salida: A, the area under the ROC curve.

$U_{sort} \leftarrow U$ sorted by decreasing values of p^i_+
$AUC \leftarrow 0$
$TP \leftarrow 0, FP \leftarrow 0$
$p_{prev} \leftarrow 0$
$tpr_{prev} \leftarrow 0, fpr_{prev} \leftarrow 0$
for $i = 1,\ldots,t$ **do**
  if $p^i_+ \neq p_{prev}$ **then**
    $tpr_{new} \leftarrow \frac{TP}{p}$
    $fpr_{new} \leftarrow \frac{FP}{n}$
    $area \leftarrow \frac{(tpr_{prev}+tpr_{new})(fpr_{new}-fpr_{prev})}{2}$
    $AUC \leftarrow AUC + area$
    $tpr_{prev} \leftarrow tpr_{new}, fpr_{prev} \leftarrow fpr_{new}$
    $p_{prev} \leftarrow p^i_+$
  end if
  if $l(x_i) = P$ **then**
    $TP \leftarrow TP + 1$
  else
    $FP \leftarrow FP + 1$
  end if
end for
ENDFOR
$AUC \leftarrow AUC + \frac{(tpr_{prev}+1)(1-fpr_{prev})}{2}$
Return AUC

de instancias por clase permaneciera uniforme Batista et al. (2004). Los conjuntos de datos particionados están disponibles en la página web de KEELAlcalá et al. (2009, 2010) (KEEL-dataset), en el hipervínculo: http://www.keel.es/datasets.php.

Para el estudio experimental, además de considerar los 102 conjuntos de datos como un todo, consideramos 3 subconjuntos de éstos atendiendo a su IR. El objetivo de esta división es evaluar el comportamiento de los algoritmos para diferentes niveles de desbalance.

1. $IR < 9$ (desbalance bajo): Este grupo contiene 22 conjuntos de datos, todos con IR menor que 9.
2. $IR \geq 9$ (desbalance alto): Este grupo contiene 80 conjuntos de datos, todos con IR de al menos 9.
3. $IR \geq 33$ (desbalance muy alto): Este grupo contiene 31 conjuntos de datos, todos con IR de al menos 33.

4.3.3. Parámetros para IFROWANN

Definición 7 *Relación de separabilidad. Con el objetivo de determinar la separabilidad aproximada entre dos instancias x e y basada en un conjunto de atributos $\mathscr{A}$, en esta investigación asumimos la siguiente definición. Dado un atributo cuantitativo a (ej. real),*

$$R_a(x,y) = 1 - \frac{|a(x) - a(y)|}{range(a)} \tag{4.25}$$

mientras que para un atributo nominal a,

$$R_a(x,y) = \begin{cases} 1 & if\ a(x) = a(y) \\ 0 & en\ otro\ caso \end{cases} \tag{4.26}$$

Consideramos las siguientes tres formas alternativas para definir las relaciones difusas R:

Tabla 4.1: Descripción de los conjuntos de datos utilizados en el estudio experimental

Dataset	IR	Inst	Attr	Dataset	IR	Inst	Attr
glass1	1,82	214	9	ecoli4	15,8	336	7
ecoli-0vs1	1,86	220	9	page-blocks-1-3vs4	15,86	472	10
wisconsinImb	1,86	683	7	abalone9-18	16,4	731	8
iris0	2	150	4	glass-0-1-6vs5	19,44	184	9
glass0	2,06	214	9	shuttle-c2-vs-c4	20.5	129	9
yeast1	2,46	1484	8	cleveland-4	21,85	297	13
habermanImb	2,78	306	3	shuttle-6vs2-3	22	230	9
vehicle2	2,88	846	18	yeast-1-4-5-8vs7	22,1	693	8
vehicle1	2,9	846	18	ionosphere-bredvsg	22,5	235	33
vehicle3	2,99	846	18	glass5	22,78	214	9
glass-0-1-2-3vs4-5-6	3,2	214	9	yeast-2vs8	23,1	482	8
vehicle0	3,25	846	18	wdbc-MredBvsB	23,8	372	30
ecoli1	3,36	336	7	texture-2redvs3-4	23,81	1042	40
appendicitisImb	4,05	106	7	yeast4	28,1	1484	8
new-thyroid1	5,14	215	5	winequalityred-4	29,17	1599	11
new-thyroid2	5,14	215	5	kddcup-guess-passwdvssatan	29,98	1642	41
ecoli2	5,46	336	7	yeast-1-2-8-9vs7	30.57	947	8
segment0	6,02	2308	19	abalone-3vs11	32,47	502	8
glass6	6,38	214	9	winequalitywhite-9vs4	32,6	168	77
yeast3	8,1	1484	8	yeast5	32,73	1484	8
ecoli3	8,6	336	7	winequalityred-8vs6	35,44	656	11
page-blocks0	8,79	5472	10	ionosphere-bredBvsg	37,5	231	33
ecoli-0-3-4vs5	9	200	7	ecoli-0-1-3-7vs2-6	39,14	281	7
ecoli-0-6-7vs3-5	9,09	222	7	abalone-17vs7-8-9-10	39,31	2338	8
yeast-2vs4	9,1	515	7	abalone-21vs8	40.5	581	8
ecoli-0-2-3-4vs5	9,1	202	7	yeast6	41,4	1484	8
glass-0-1-5vs2	9,12	172	9	segment-7redvs2-4-5-6	42,58	1351	19
yeast-0-3-5-9vs7-8	9,12	506	8	winequalitywhite-3vs7	44	900	11
yeast-0-2-5-6vs3-7-8-9	9,14	1004	8	wdbc-MredvsB	44,63	365	30
yeast-0-2-5-7-9vs3-6-8	9,14	1004	8	segment-5redvs1-2-3	45	1012	19
ecoli-0-4-6vs5	9,15	203	6	winequalityred-8vs6-7	46,5	855	11
ecoli-0-1vs2-3-5	9,17	244	7	phoneme-1redvs0red	46,98	2543	5
ecoli-0-2-6-7vs3-5	9,18	224	7	texture-6redvs7-8	47,62	1021	40
glass-0-4vs5	9,22	92	9	kddcup-landvsportsweep	49,52	1061	41
ecoli-0-3-4-6vs5	9,25	205	7	abalone-19vs10-11-12	49,69	1622	8
ecoli-0-3-4-7vs5-6	9,28	257	7	magic-hredvsgred	54,1	2645	10
yeast-0-5-6-7-9vs4	9,35	528	8	winequalitywhite-3-9vs5	58,28	1482	11
ecoli-0-6-7vs5	10	220	6	shuttle-2vs5	66,67	3316	9
glass-0-1-6vs2	10.29	192	9	winequalityred-3vs5	68,1	691	11
ecoli-0-1-4-7vs2-3-5-6	10.59	336	7	phoneme-1redBvs0redB	69,7	2333	5
ecoli-0-1vs5	11	240	6	texture-12redvs13-14	71,43	1014	40
glass-0-6vs5	11	108	9	abalone-20vs8-9-10	72,69	1916	8
glass-0-1-4-6vs2	11,06	205	9	kddcup-bufferoverowvsback	73,43	2233	41
glass2	11,59	214	9	kddcup-landvssatan	75,67	1610	41
ecoli-0-1-4-7vs5-6	12,28	332	7	shuttle-2vs1red	81,63	4049	9
cleveland-0vs4	12,31	173	13	segment-6redvs3-4-5	82,5	1002	19
ecoli-0-1-4-6vs5	13	280	6	shuttle-6-7vs1red	86,96	2023	9
movement-libras-1	13	336	90	magic-hredBvsgredB	88	2403	10
shuttle-c0-vs-c4	13,87	1829	9	texture-7redvs2-3-4-6	95,24	2021	40
yeast-1vs7	14,3	459	7	kddcup-rootkit-imapvsback	100.14	2225	41
glass4	15,46	214	9	abalone19	129,44	4174	8

$$R_{Min}(x,y) = \text{mín}(R_{a_1}(x,y),\ldots,R_{a_m}(x,y)) \quad (4.27)$$

$$R_{\mathscr{T}_L}(x,y) = \mathscr{T}_L(R_{a_1}(x,y),\ldots,R_{a_m}(x,y)) \quad (4.28)$$

$$R_{Av}(x,y) = \frac{R_{a_1}(x,y)+\ldots+R_{a_m}(x,y)}{m} \quad (4.29)$$

donde la t-normŁ ukasiewicz $\mathscr{T}_L$ está definida por, para $u_1,u_2,\ldots,u_m$ en $[0,1]$,

$$\mathscr{T}_L(u_1,u_2,\ldots,u_m) = \text{máx}(u_1+u_2+\ldots+u_m-m,0) \quad (4.30)$$

Vectores de peso: basándonos en las propuestas hechas en la sección 4.2.2, consideramos las siguientes 6 configuraciones de pesos para los vectores OWA:

1. $\mathscr{W}_1 = \langle W_P^{l_1}, W_N^{l_1}\rangle$
2. $\mathscr{W}_2 = \langle W_P^{l_1}, W_N^{l_2}\rangle$
3. $\mathscr{W}_3 = \langle W_P^{l_2}, W_N^{l_1}\rangle$
4. $\mathscr{W}_4 = \langle W_P^{l_2}, W_N^{l_2}\rangle$
5. $\mathscr{W}_5 = \langle W_P^{l_1,\gamma}, W_N^{l_1}\rangle$ with $\gamma = 0{,}1$
6. $\mathscr{W}_6 = \langle W_P^{l_1,\gamma}, W_N^{l_2}\rangle$ with $\gamma = 0{,}1$

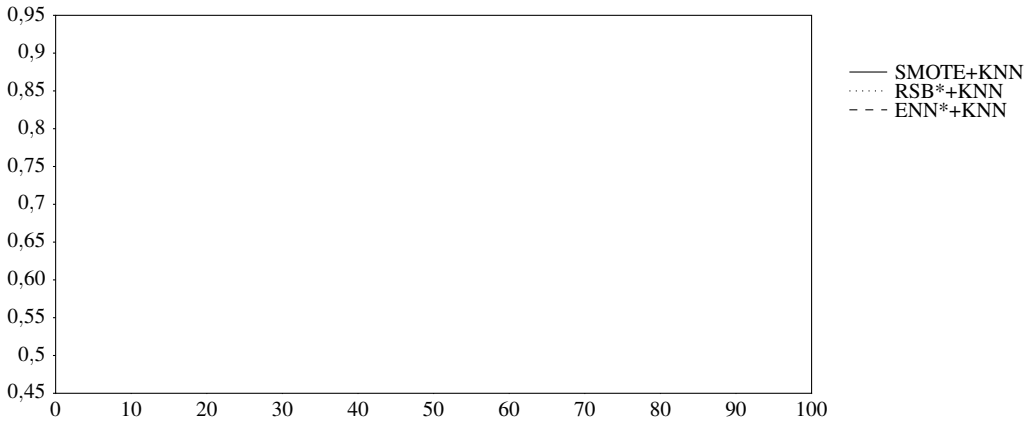
En la siguiente sección realizaremos un estudio de sensibilidad al parámetro γ para las dos últimas propuestas, con el objetivo de evaluar si el algoritmo FROWA es robusto antes variaciones en el γ.

4.3.4. Algoritmos seleccionados del estado del arte para comparar

Para el estudio experimental se consideraron los siguientes algoritmos para comparar:

- SMOTE: usando como clasificador kNN, C4.5 y SVM
- SMOTE-RSB*: usando como clasificador kNN, C4.5 y SVM

Figura 4.2: Media de AUC obtenida para 100 valores equidistantes de k para kNN para los 3 algoritmos de preprocesamientos usados en nuestro estudio



- SMOTE-ENN: usando como clasificador kNN, C4.5 y SVM
- CS-C4.5
- CS-SVM
- EUSBoost

Los tres primeros algoritmos seleccionados son algoritmos de preprocesamiento basados en SMOTE (descritos en la sección 1.3.1) combinados con 3 de los clasificadores más usados (kNN, C4.5 y SVM), lo que nos lleva a tener 9 variantes con las que comparar. Los siguientes dos algoritmos CS-C4.5 y CS-SVM (sección 1.3.3)son algoritmos con aprendizaje sensible al coste, el primero basado en el árbol de decisión C4.5 y el segundo en una máquina de soporte vectorial SVM. Finalmente seleccionamos el ensemble EUSBOOST (descrito en 1.3.2).

Para el algoritmo kNN se realizaron 100 ejecuciones para distintos valores de k (100 puntos equidistantes entre 1 y la totalidad de instancias en el conjunto de entrenamiento). La figura 4.2 muestra la media alcanzada por los algoritmos para cada valor de k. Para nuestros experimentos se seleccionó el mejor resultado.

4.4. Resultados experimentales

En esta sección presentamos el resultado del análisis experimental [2]. El estudio experimental lo hemos divido en 2 partes: primero compararemos el algoritmo VCCAD con los 18 algoritmos IFROWANN resultantes de combinar las 3 relaciones de separabilidad con las 6 estrategias de construcción de vectores OWA, de esta primera, valiéndonos de pruebas estadísticas seleccionaremos las 3 propuestas más competitivas.En particular, determinamos cuáles son las propuestas más competitivas en cada uno de los cuatro bloques de experimentos considerados (todos los conjuntos, los de bajo IR, los de IR alto y los de IR muy alto). Segundo: compararemos las propuestas que mejor resultaron en el primer análisis con los algoritmos del estado del arte que hemos seleccionado para el estudio comparativo.

4.4.1. Resultados experimentales de los algoritmos propuestos

La tabla 4.2 muestra la media alcanzada por los algoritmos propuestos para cada uno de los bloques, en el Apéndice C se muestran los resultados detallados.

[2]Los resultados detallados, para cada algoritmo para cada conjunto de datos, se encuentran disponible en el sitio asociado a este trabajo, http://sci2s.ugr.es/frowa-imbalanced/

Tabla 4.2: Media de AUC para las variantes de IFROWANN/VCCAD. Los valores marcados en azul (mayores que 0.91) y los marcados en verde (mejor valor para cada variante de VCCAD) son los que tuvieron en cuenta para el análisis estadístico.

Method	all	<9	≥9	≥33
TL-$\mathscr{W}_1$	0.8943	0.9186	0.8877	0.8845
TL-$\mathscr{W}_2$	0.8802	0.9076	0.8727	0.8424
TL-$\mathscr{W}_3$	0.8893	0.8935	0.8882	0.8941
TL-$\mathscr{W}_4$	0.8998	0.9180	0.8948	0.8959
TL-$\mathscr{W}_5$	0.8928	0.9163	0.8863	0.8925
TL-$\mathscr{W}_6$	0.9054	0.9148	0.9028	0.8989
AV-$\mathscr{W}_1$	0.9098	0.9014	0.9121	0.9023
AV-$\mathscr{W}_2$	0.9094	0.9029	0.9112	0.8938
AV-$\mathscr{W}_3$	0.8990	0.8900	0.9014	0.8938
AV-$\mathscr{W}_4$	0.9181	**0.9232**	0.9167	0.9073
AV-$\mathscr{W}_5$	0.9122	0.9068	0.9136	0.9030
AV-$\mathscr{W}_6$	**0.9256**	0.9139	**0.9288**	0.9166
MIN-$\mathscr{W}_1$	0.8936	0.8844	0.8961	0.9062
MIN-$\mathscr{W}_2$	0.8908	0.8809	0.8935	0.8990
MIN-$\mathscr{W}_3$	0.8813	0.8713	0.8841	0.8975
MIN-$\mathscr{W}_4$	0.9030	0.9101	0.9010	0.9156
MIN-$\mathscr{W}_5$	0.8955	0.8877	0.8977	0.9085
MIN-$\mathscr{W}_6$	0.9071	0.8925	0.9111	**0.9230**
VCCAD-TL	0.8905	0.9044	0.8866	0.8815
VCCAD-AV	0.9083	0.9074	0.9086	0.8974
VCCAD-MIN	0.8925	0.8992	0.8906	0.9030

En cada columna el mayor valor de AUC lo hemos resaltado en negrita. Como se observa, el algoritmo basado en AV-$\mathscr{W}_6$ obtiene la mejor media de AUC cuando consideramos los 102 conjuntos de datos, y también para los 80 conjuntos de alto desbalance ($IR \geq 9$). Sin embargo para los conjuntos de IR bajo ($IR < 9$) el algoritmo que mejor se comporta el AV-$\mathscr{W}_4$, mientras que para los conjuntos de IR muy alto ($IR \geq 33$) la mejor variante resultó MIN-$\mathscr{W}_6$.

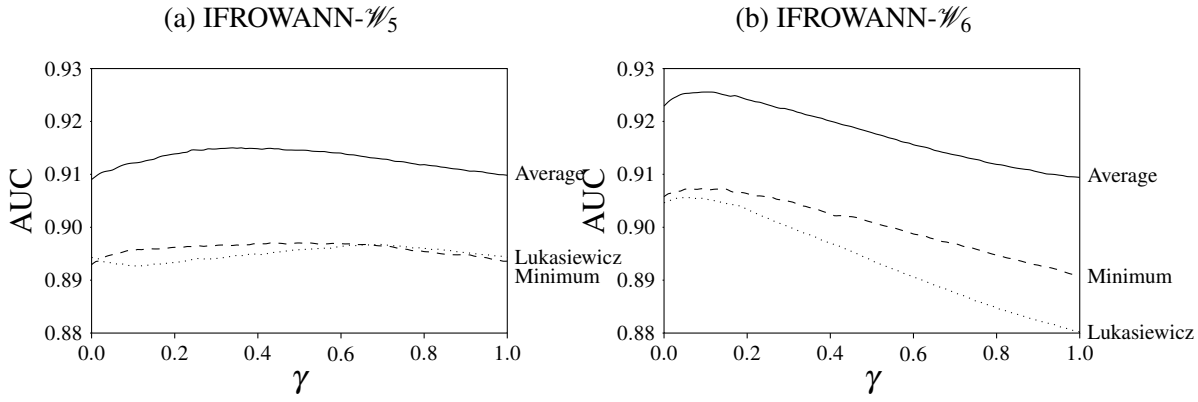
Cuando comparamos IFROWANN con VCCAD, podemos observar que las estrategias de ponderación ($\mathscr{W}_1$, $\mathscr{W}_4$, $\mathscr{W}_5$ and $\mathscr{W}_6$), generalmente funcionan mejor que la variante correspondiente de VCCAD, excepto para bajo IR cuando se usa AV y MIN, pero en estos casos $\mathscr{W}_4$ es todavía mejor que el algoritmo VCCAD básico.

Existe una diferencia medianamente notable entre los resultados obtenidos usando cada uno de las relaciones de inseparabilidad (TL, AV y MIN). En general los resultados son obtenidos cuando se usa AV, incluyendo el algoritmo básico VCCAD. TL funciona mejor para conjuntos de bajo desbalance, sin lograr la mejor media en AUC. Por otra parte para IR muy alto, el mejor resultado se logra usando MIN como relación de inseparabilidad.

Entre las estrategias de ponderación existen también marcadas diferencias:

- Decrementar los pesos de manera exponencial $\mathscr{W}_4$ resulta mejor que decrementarlos de forma lineal $\mathscr{W}_1$.
- Mezclar diferentes estrategias de ponderación ($\mathscr{W}_2$ y $\mathscr{W}_3$) generalmente empeora los resultados comparados con $\mathscr{W}_1$ y $\mathscr{W}_4$.
- La variante $\mathscr{W}_1$ para solo ponderar una fracción de las instancias negativas ($\mathscr{W}_5$), mejora ligeramente los resultados cuando se usa AV y MIN como relación de inseparabilidad, sin embargo continúa siendo inferior a $\mathscr{W}_4$. Por otra parte, esta estrategia baja los resultados cuando se usa TL. La figura 4.3a muestra un análisis de sensibilidad para el parámetro γ, obtenido para todos los conjuntos cuando el parámetro se mueve entre 0 y 1. Se puede observar que éste muestra un comportamiento bastante estable. Para el caso de TL, la curva muestra un ligero descenso para valores pequeños de γ, lo cual pudiera explicar el hecho que se obtengan peores resultados que $\mathscr{W}_1$ en este caso.
- La variante $\mathscr{W}_4$ para solo ponderar una fracción de las instancias negativas ($\mathscr{W}_6$) mejora los resultados para conjuntos del IR alto, pero los empeora ligeramente para IR bajo. El análisis de sensibilidad a γ se muestra en la figura 4.3b, donde se observa que los mejores resultados se logran para valores pequeños de gamma, lo cual justifica nuestra elección de seleccionar $\gamma = 0{,}1$.

Figure 4.3: Análisis de sensibilidad a γ para las estrategias de ponderación $\mathscr{W}_5$ y $\mathscr{W}_6$ evaluado sobre todos los conjuntos de datos



Seguidamente procedemos con el análisis estadístico de los resultados para cada uno de los bloques de experimentos definidos. Con objetivo de reducir el número de variantes en cada prueba, y de aumentar el poder discriminatorio de las pruebas estadísticas, hemos seleccionado solo las propuestas de mejor media en AUC para cada bloque (mayores que 0.91), estos valores fueron resaltados en azul en la tabla 4.2. Consideramos además los mejores resultados del algoritmo básico VCCAD para cada bloque, resaltado en verde en la tabla.

Análisis estadístico para todos los conjuntos de datos

El promedio de ranking obtenido por los algoritmos así como el p-value ajustado obtenido por el procedimiento pos-hoc de Holm's se muestra en la tabla 4.3. El p-value obtenido por Friedman es 0. lo cual indica que la hipótesis de equivalencia puede ser rechaza con un alto nivel de confianza.

Tabla 4.3: Average Friedman rankings y p-values ajustados usando el procedimiento post-hoc Holm's para todos los conjuntos de datos, usando AV-$\mathscr{W}_6$ como algoritmo de control.

Algorithm	Average Friedman ranking	Adjusted p-value
AV-$\mathscr{W}_6$	2.0196	-
AV-$\mathscr{W}_4$	2.25	0.202498
AV-$\mathscr{W}_5$	2.6618	0.000764
VCCAD-AV	3.0686	0

Como se puede observar, AV-$\mathscr{W}_6$ obtiene el ranking más bajo, por lo cual es usado como algoritmo de control. El p-value ajustado es lo suficientemente bajo como para rechazar la hipótesis nula con alto nivel de confianza para los algoritmos AV-$\mathscr{W}_5$ y VCCAD-AV. Esta prueba nos permite concluir que el algoritmo propuesto IFROWANN, usando la estrategia de ponderación adecuada, puede superar de manera significativa el algoritmo VCCAD original.

Análisis estadístico para los conjuntos de IR bajo

Para los conjuntos de IR bajo se consideraron 8 propuestas para el análisis estadístico. La tabla 4.4 muestra el average ranking obtenido por la prueba de Friedman, como se

puede observar el algoritmo que mejor ranking obtiene es el AV-$\mathscr{W}_4$. El p-value de la prueba de Friedman es 0.049, que nos permite concluir una vez más que existen diferencias significativas entre los algoritmos comparados.

Basándonos en los p-values ajustados, la prueba post-hoc de Holm nos permite concluir que el algoritmo AV-$\mathscr{W}_4$ es significativamente mejor que el algoritmo VCCAD-AV. Los valores pequeños de p-values ajustados para AV-$\mathscr{W}_6$ y TL-$\mathscr{W}_4$ nos indican que en este caso la estartegia $\mathscr{W}_4$ es mejor que $\mathscr{W}_6$.

Tabla 4.4: Average Friedman rankings y p-values ajustados usando el procedimiento post-hoc Holm's para los conjuntos de datos de bajo IR, usando AV-$\mathscr{W}_4$ como algoritmo de control.

Algorithm	Average Friedman ranking	Adjusted p-value
AV-$\mathscr{W}_4$	3.0682	-
TL-$\mathscr{W}_4$	3.7955	0.324756
AV-$\mathscr{W}_6$	4.4773	0.117685
TL-$\mathscr{W}_6$	4.5909	0.117685
TL-$\mathscr{W}_1$	4.6818	0.115592
TL-$\mathscr{W}_5$	4.9545	0.053224
VCCAD-AV	5.0909	0.037
MIN-$\mathscr{W}_4$	5.3409	0.014623

Análisis estadístico para los conjuntos de IR alto

La tabla 4.5 muestra el average ranking obtenido por la prueba de Friedman para las 7 propuestas seleccionadas en este caso. El p-value obtenido por la prueba de Friedman es 0. lo cual indica que podemos rechazar la hipótesis de equivalencia con un alto nivel de confianza. Como se pude observar, el mejor ranking es obtenido por AV-$\mathscr{W}_6$, por lo cual es usado como algoritmo de control. Los valores de p-values ajustados son todos muy pequeños, lo cual indica que el algoritmo AV-$\mathscr{W}_6$ es significativamente mejor que las restantes propuestas cuando se enfrentan a conjuntos de alto IR.

Tabla 4.5: Average Friedman rankings y p-values ajustados usando el procedimiento post-hoc Holm's para los conjuntos de datos de alto IR, usando AV-$\mathscr{W}_6$ como algoritmo de control

Algorithm	Average Friedman ranking	Adjusted p-value
AV-$\mathscr{W}_6$	2.6	-
AV-$\mathscr{W}_4$	3.7	0.00128
AV-$\mathscr{W}_5$	3.825	0.00067
AV-$\mathscr{W}_2$	4.1688	0.000013
AV-$\mathscr{W}_1$	4.3812	0.000001
MIN-$\mathscr{W}_6$	4.6562	0
VCCAD-AV	4.6688	0

Análisis estadístico para los conjuntos de IR muy alto

La tabla 4.6 muestra el average ranking obtenido por la prueba de Friedman para las 4 propuestas seleccionadas en este caso. El p-value obtenido por la prueba de Friedman es 0.065, lo cual indica que podemos rechazar la hipótesis de equivalencia con un alto nivel de confianza. Como se pude observar, el mejor ranking es obtenido por AV-$\mathscr{W}_6$, por lo cual es usado como algoritmo de control. Sin embargo esta vez el procedimiento post-hoc de Holm no encontró diferencias significativas entre la muestra de control y los algoritmos MIN-$\mathscr{W}_4$ y MIN-$\mathscr{W}_6$, por otra parte el algoritmo de control no superó al algoritmo VCCAD-MIN.

Tabla 4.6: Average Friedman rankings y p-values ajustados usando el procedimiento post-hoc Holm's para los conjuntos de datos de muy alto IR, usando AV-$\mathscr{W}_6$ como algoritmo de control.

Algorithm	Average Friedman ranking	Adjusted p-value
AV-$\mathscr{W}_6$	2.2581	-
MIN-$\mathscr{W}_6$	2.3065	1.315991
MIN-$\mathscr{W}_4$	2.4032	1.315991
VCCAD-MIN	3.0323	0.054681

4.4.2. Análisis comparativo con los algoritmos del estado del arte

El estudio experimental desarrollado en la sección anterior nos mostró que las dos mejores propuestas son AV-$\mathscr{W}_4$ para el caso del IR bajo y AV-$\mathscr{W}_6$ para el resto de los casos. En esta sección se comparan estas dos propuestas con los algoritmos seleccionados del estado del arte. Los resultados de la media de AUC para los bloques se muestran en la tabla I.1.

Tabla 4.7: Media de AUC para los algoritmos del estado del arte y las mejores variantes de IFROWANN. Los valores marcados en azul (mayores que 0.9) son los que se tendrán en cuenta para el análisis estadístico.

Method	all	<9	>9	>33
SMOTE-kNN	0.9096	0.9143	0.9083	0.8987
SMOTE-C4.5	0.8315	0.8604	0.8235	0.8050
SMOTE-SVM	0.9000	0.9051	0.8986	0.9133
SMOTE-ENN-kNN	0.8839	0.9093	0.8769	0.8320
SMOTE-ENN-C4.5	0.8412	0.8714	0.8329	0.8218
SMOTE-ENN-SVM	0.9005	0.9046	0.8994	0.9130
C4.5-CS	0.8263	0.8691	0.8146	0.8083
SVM-CS	0.8952	0.9137	0.8901	0.9032
EUSBOOST	0.9094	**0.9263**	0.9048	0.8977
SMOTE-RSB$_*$-kNN	0.9085	0.9119	0.9076	0.8975
SMOTE-RSB$_*$-C4.5	0.8266	0.8681	0.8152	0.8021
SMOTE-RSB$_*$-SVM	0.9001	0.9036	0.8991	0.9130
AV-$\mathscr{W}_4$	0.9181	0.9232	0.9167	0.9073
AV-$\mathscr{W}_6$	**0.9256**	0.9139	**0.9288**	**0.9166**

Como se puede observar $\mathscr{W}_6$ obtiene la mejora media para todos los casos excepto para IR bajo donde EUSBOOST logra la mejor media. Una vez más someteremos estos resultados a pruebas estadísticas. En este caso, para cada bloque solo consideraremos los algoritmos que hayan obtenido $AUC \geq 0{,}9$, (marcados en azul en la tabla I.1).

Análisis estadístico para todos los conjuntos

La tabla 4.8 muestra el average ranking obtenido por una prueba de Friedman. El p-value de la prueba de Friedman es 0.000006, lo cual indica que la hipótesis de equivalencia puede ser rechazada con un alto nivel de confianza. Como se puede observar, el mejor ranking lo obtiene el algoritmo AV-$\mathscr{W}_6$. Los p-values ajustados demuestran que el algoritmo AV-$\mathscr{W}_6$ es estadísticamente superior al resto de los algoritmos.

Tabla 4.8: Average Friedman rankings y p-values ajustados usando el procedimiento post-hoc Holm's para todos los conjuntos de datos, usando AV-$\mathscr{W}_6$ como algoritmo de control.

Algorithm	Average Friedman ranking	Adjusted p-value
AV-$\mathscr{W}_6$	2.9363	-
SMOTE-RSB$_*$-kNN	3.9461	0.000843
SMOTE-kNN	4.0343	0.00075
EUSBOOST	4.0441	0.00075
SMOTE-RSB$_*$-SVM	4.201	0.000116
SMOTE-SVM	4.2794	0.000045
SMOTE-ENN-SVM	4.5588	0

Análisis estadístico para los conjuntos de IR bajo

La tabla 4.9 muestra el ranking obtenido por la prueba de Friedman. En este caso el p-value asociado a dicha prueba es de 0.20568, el cual no es lo suficientemente bajo como para rechazar la hipótesis de equivalencia, por lo que podemos concluir que no existen diferencias significativas entre los algoritmos comparados. Nótese que la mejor media en AUC del bloque la obtuvo el algoritmo EUSBOOST, sin embargo el mejor ranking fue obtenido por AV-$\mathscr{W}_4$.

Tabla 4.9: Average Friedman rankings para conjuntos de IR bajo. La prueba de Friedman no encuentra diferencias significativas entre los algoritmos, por tanto la prueba de Holm no se realiza.

Algorithm	Average Friedman ranking
AV-$\mathscr{W}_4$	3.5455
SVM-CS	4.4773
EUSBOOST	4.75
SMOTE-kNN	5.0227
SB-kNN	5.1591
SMOTE-SVM	5.2045
SB-SVM	5.3864
SMOTE-ENN-kNN	5.7273
SMOTE-ENN-SVM	5.7273

Análisis estadístico para los conjuntos de IR alto

Los resultados que se muestran en la tabla 4.10 son similares a los obtenidos para todos los conjuntos de datos. El p-value obtenido por la prueba de Friedman es 0. AV-$\mathscr{W}_6$ obtiene el mejor ranking con diferencias altamente significativas con respecto al resto de los algoritmos.

Tabla 4.10: Average Friedman rankings y p-values ajustados usando el procedimiento post-hoc Holm's para los conjuntos de datos del alto IR, usando AV-$\mathscr{W}_6$ como algoritmo de control.

Algorithm	Average Friedman ranking	Adjusted p-value
AV-$\mathscr{W}_6$	1.6875	-
SMOTE-RSB$_*$-kNN	2.6875	0.0
EUSBOOST	2.7562	0.0
SMOTE-kNN	2.8687	0.000001

Análisis estadístico para los conjuntos de IR muy alto

En este caso, el p-value obtenido por la prueba de Friedman es 0.656806, lo cual indica que no existen diferencias significativas entre los algoritmos comparados. En la tabla 4.11 se muestra el ranking obtenido por la prueba de Friedman, resulta interesante ver cómo SMOTE-RSB*-SVM obtiene el mejor ranking mientras que la media más alta fue obtenida por AV-$\mathscr{W}_6$.

Tabla 4.11: Average Friedman rankings para conjuntos de IR muy alto. La prueba de Friedman no encuentra diferencias significativas entre los algoritmos, por tanto la prueba de Holm no se realiza.

Algorithm	Average Friedman ranking
SMOTE-RSB*-SVM	2.7419
SMOTE-SVM	2.8387
SMOTE-ENN-SVM	3.0161
SVM-CS	3.0968
AV-$\mathscr{W}_6$	3.3065

Para complementar el estudio estadístico, realizamos un análisis gráfico del comportamiento de nuestras dos mejores propuestas AV-$\mathscr{W}_6$ y AV-$\mathscr{W}_4$ y los algoritmos que resultaron más competitivos dentro del estado del arte.

La figura 4.4 muestra por el AUC obtenido para todos los conjuntos (eje Y) ordenado por el eje X atendiendo a su IR. De manera similar en la figura J.2, mostramos el análisis de forma más detallada para cada uno de los bloques de experimentos realizados, considerando en cada caso los mejores algoritmos del estado del arte.

En ambas figuras podemos observar que tanto para IR bajo como para muy alto, los algoritmos comparados se comportan de manera más o menos similar, mientras que las diferencias más marcadas ocurren en el rango de IR entre 9 y 33, donde el algoritmo AV-$\mathscr{W}_6$ obviamente es el de mejor comportamiento.

Figura 4.4: AUC para todos los conjuntos de datos ordenados atendiendo a su IR, para nuestra mejor propuesta (AV-$\mathscr{W}_6$) y los algoritmos más competitivos del estado del arte (SMOTE-RSB$_*$-kNN y EUSBOOST)a

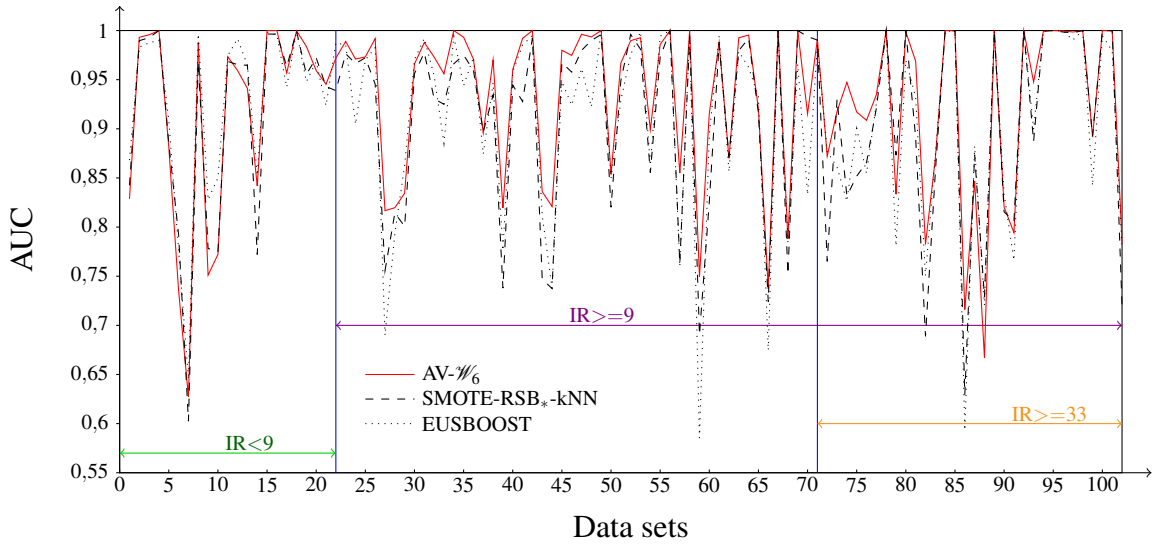
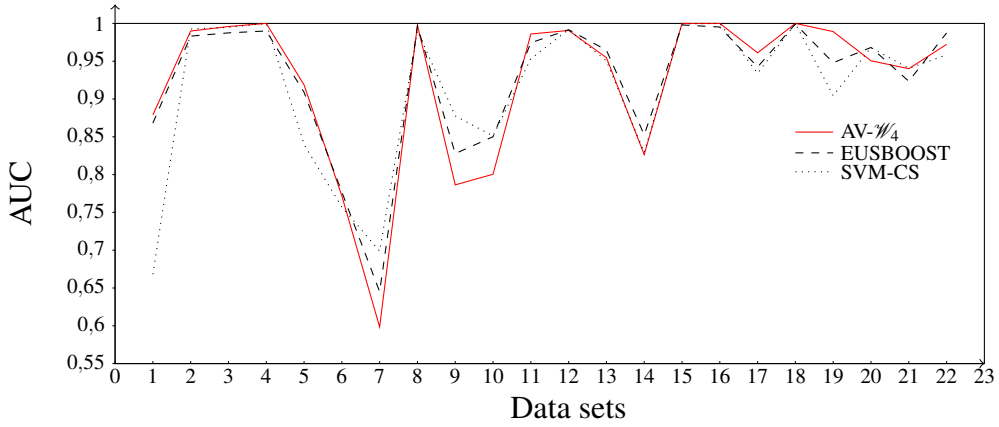
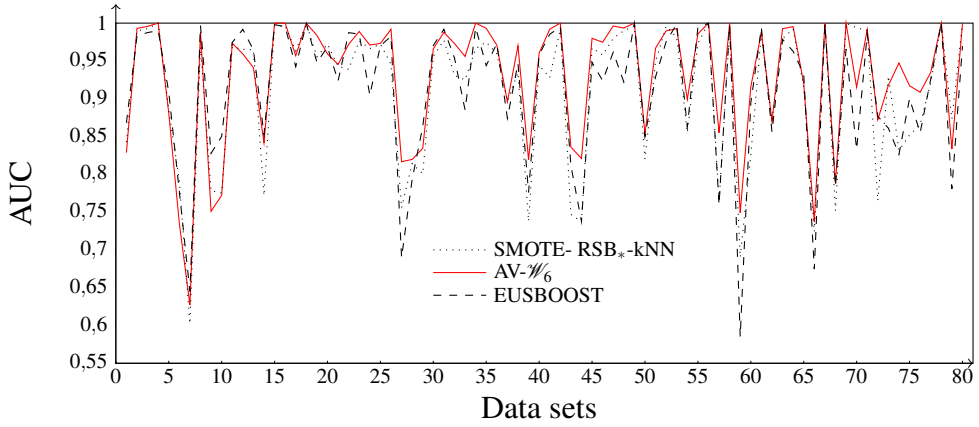


Figura 4.5: AUC para todos los bloques de datos. Los conjuntos están ordenados atendiendo a su IR.

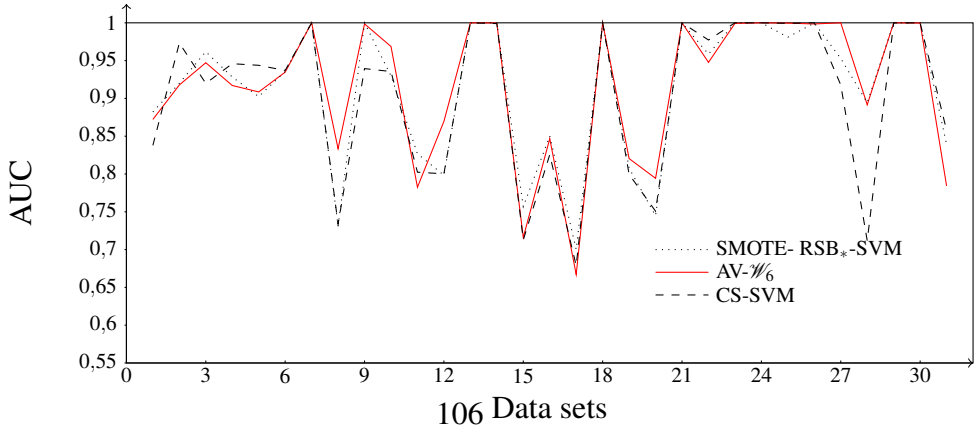
(a) **IR<9** Low Imbalance



(b) **IR>=9** High Imbalance



(c) **IR>=33** Very High Imbalance



4.5. Conclusiones parciales

En este capítulo hemos presentado una nueva propuesta para clasificación de conjuntos de datos no balanceados. La propuesta introduce el uso de los vectores OWA (Ordered Weighted Average) en el algoritmo base Vecinos más Cercanos con Conjuntos Aproximados Difusos (VCCAD). Se proponen además 6 estrategias de construcción de vectores OWA considerando en todos los casos la baja representatividad de una de las clases, estas 6 estrategias son combinadas con 3 formas diferentes de determinar las relaciones de separabilidad (Lukasiewicz, Average y Minimum).

La novedad de nuestra propuesta radica en el uso de la hibridación difusa de la teoría de los conjuntos aproximados (Fuzzy Rough Set Theory), específicamente el uso del grado de pertenencia a la región positiva de cada instancia a ambas clases, para clasificarlas en positivas o negativas. El uso de operadores OWA nos permite tener en cuenta el grado de desbalance a la hora de calcular la pertenencia de una instancia dada a la región positiva de una clase.

Finalmente, después del estudio experimental y el análisis estadístico realizado podemos concluir que nuestra propuesta resulta muy competitiva en dominios no balanceados, pues logran resultados significativamente superiores a 12 algoritmos del estado del arte.

Comentarios finales

"Research is what I'm doing when I don't know what I'm doing."
– Wernher Von Braun

A modo de conclusiones se presenta en este capítulo los principales resultados y aportes de la labor de investigación que se recoge en esta tesis. El trabajo realizado no solo ha permitido obtener buenos resultados con los algoritmos propuestos, sino también resolver problemas de ingeniería de manera eficiente. Quedan abiertas además nuevas líneas e ideas a desarrollar que proponemos como trabajo futuro.

Conclusiones

En esta investigación nos hemos centrado en los algoritmos de preprocesamiento para conjuntos de datos no balanceados. Del estudio de los algoritmos del estado del arte se detectó que los algoritmos de limpieza que se utilizan no resultan del todo eficiente, debido a que la técnica usada no garantiza que todos los ejemplos sintéticos que se generen pertenezcan con absoluta certeza a su clase.

Por otra parte algunos algoritmos del estado del arte tienen tendencia a eliminar ejemplos originales usando criterios bajo los cuales pudieran ser eliminados ejemplos muy útiles para la etapa de clasificación, en esta investigación se ha demostrado que la eliminación de ejemplos originales puede ser beneficiosa si se hace bajo criterios menos restrictivos que los usados para los ejemplos originales y si no elimina ningún ejemplo de la clase minoritaria.

Con el estudio que se realiza en esta tesis de la teoría de los conjuntos aproximados y la teoría de los conjuntos aproximados difusa, se pudo constatar que la utilización del concepto de la aproximación inferior resulta de suma utilidad para la selección de las instancias más representativas de su clase.

1. La TCA permite decantar cuáles ejemplos pertenecen con absoluta certeza a su clase y cuáles están en duda (en la región frontera). Valiéndonos de la utilidad de este concepto se propone el algoritmo **SMOTE-RSB***, que genera ejemplos sintéticos usando SMOTE y luego chequea la calidad de estos evaluando su pertenencia a la región positiva de su clase, este proceso es repetido hasta que ambas clases queden igualadas o hasta que se alcance un umbral de similaridad alto. El estudio experimental realizado muestra la robustez del algoritmo frente a 44 conjuntos de datos de alto índice de desbalance en comparación con 6 reconocidos algoritmos del estado del arte, y valiéndonos de pruebas no paramétricas para mostrar la superioridad estadística.

   Con el uso de la aproximación inferior difusa de la TCA se logra cuantificar de manera gradual la pertenencia de los ejemplos a la región positiva de su clase; de esta forma es también posible editar la clase mayoritaria con menor riesgo de eliminar demasiados ejemplos originales que pudieran afectar el resultado de la clasificación.
2. Usando este principio se propone el algoritmo **SMOTE-TCAD**, un algoritmo que selecciona solo los ejemplos sintéticos o los mayoritarios que tengan un grado de pertenencia a la región positiva por encima de un umbral dado. Este algoritmo permitió además de seleccionar ejemplos sintéticos de alta calidad, poder editar a la vez los ejemplos mayoritarios originales quedándonos únicamente con los más representativos. El estudio experimental realizado demostró la superioridad de la propuesta por encima de 6 conocidos algoritmos del estado del arte. Esta propuesta pone de manifiesto lo beneficioso que resulta el uso de la TCAD en la selección de instancias.

   Una nueva propuesta **SMOTE-TCAD-2T** es realizada como extensión de SMOTE-TCAD. Este nuevo algoritmo permite la edición por separado de los ejemplos positivos y los negativos. El algoritmo genera ejemplos sintéticos usando SMOTE, luego se procede a realizar la limpieza usando 2 umbrales diferentes: uno muy

elevado para los ejemplos sintéticos; y otro muy pequeño para los ejemplos de la clase negativa. Nuestro objetivo es que los ejemplos originales que se generen sean de alta calidad, para ello solo se seleccionan los que pertenezcan a la región positiva difusa de su clase con grado igual 1; por otra parte son eliminados del conjunto de entrenamiento final los ejemplos de la clase mayoritaria cuya pertenencia a la región positiva difusa de su clase esté por debajo de 0.03. Con el uso de esta propuesta se logra obtener un conjunto de entrenamiento final con ejemplos sintéticos de alta calidad, y con una clase mayoritaria sin ejemplos que pudieran ser considerados como ruido y afectar la etapa de clasificación.

Otro aporte de nuestra investigación es la aplicación del algoritmo SMOTE-TCAD-2T para resolver el problema del diagnóstico de la necesidad de mantenimiento de los interruptores de alta potencia. Esta aplicación demostró la robustez del algoritmo al ser capaz de lograr mejores resultados que 9 algoritmos del estado del arte, logrando reducir de manera significativa tanto el número de falsos positivos como de falsos negativos.

3. Finalmente hemos propuesto un nuevo algoritmo de clasificación para conjuntos no balanceados. El algoritmo **IFROWANN** constituye una extensión del algoritmo VCCAD, solo que nuestra propuesta usa el operador OWA para construir vectores de ponderación que permitan tener en cuenta la distribución de ejemplos por clases a la hora de calcular las aproximaciones.

   IFROWANN para predecir la clase de una nueva instancia del conjunto de prueba x, calcula la suma de los grados de pertenencia de x a la aproximación inferior y superior de cada clase, y asigna la instancia a la clase cuya suma haya sido mayor. Para determinar el grado de pertenencia a las regiones positiva y negativa se utiliza el operador OWA.

   En esta tesis de diseñan 6 configuraciones diferentes para construir los vectores de pesos, y se utilizan 3 relaciones de separabilidad: la t-norma Lukasiewicz, el Mínimo y el Average; de esta forma se logran 18 nuevas variantes del algoritmo IFROWANN. El estudio experimental es realizado con 102 conjuntos de datos de diferentes índices de desbalance, para un mejor análisis del comportamiento de la propuesta el análisis de los resultados se realiza en 4 grupos: usando los 102 conjuntos, usando solo los de desbalance bajo (menor que 9), usando solo los de desbalance alto (mayor que 9)

y usando los de desbalance muy alto (mayor que 33). Los resultados de las pruebas estadísticas demuestran la superioridad del algoritmo propuesto por encima de 12 algoritmos del estado del arte.

Publicaciones asociadas a la tesis

A continuación se presenta un listado de las publicaciones obtenidas asociadas a esta tesis:

(a) Publicados/sometidos en revistas internacionales:

1. E. Ramentol, Y. Caballero, R. Bello y F. Herrera (2012). SMOTE-RSB*: A hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory. *International Journal of Knowledge and Information Systems.* 33:245–265.
   a) Factor de Impacto: 2.225.
   b) Estado: Publicado.
2. E. Ramentol, I. Gondres, S. Lajes, Y. Caballero, R. Bello, C. Cornelis, F. Herrera. Fuzzy-Rough Imbalanced Learning for the Diagnosis of High Voltage Circuit Breaker Maintenance: the SMOTE-TCAD-2T Algorithm. Sometido a la revista *Engineering Applications of Artificial Intelligence*, (2013).
   a) Factor de Impacto: 1.625.
   b) Estado: Sometido a revisión.
3. E. Ramentol, S. Vluymans, N. Verviest, Y. Caballero, R. Bello C. Cornelis, F. Herrera. IFROWANN: Imbalanced Fuzzy-Rough Ordered Weighted Average Nearest Neighbor Classification. Sometido a la revista *IEEE Transactions on Fuzzy Systems.(2014)*
   a) Factor de Impacto: 5.484.
   b) Estado: Sometido a revisión.

(b) En congresos internacionales:

1. E. Ramentol, N. Verbiest, R. Bello, Y. Caballero, Chris Cornelis and F. Herrera. (2012). SMOTE-TCAD: A new resampling method using fuzzy rough set theory. Proceedings of the 10th International FLINS Conference on uncertainty Modeling in Knowledge Engineering and Decision Making (FLINS2012). p 800-805. Estambul, Turquía.

2. E. Ramentol, Y. Sánchez, R. Bello, Y. Caballero y F. Herrera.. Edición de Conjuntos de Entrenamiento no balancedos haciendo uso de Operadores Genéticos y de la Teoría de los Conjuntos Aproximados. VI Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, MAEB´09. ISBN: 978-84-691-6813-4. Málaga.

Premios y reconocimientos asociados a la tesis

1. PREMIO DE LA ACADEMIA DE CIENCIAS DE CUBA 2014. Contribuciones al aprendizaje automatizado a través de la Teoría de los Conjuntos Aproximados Extendida.
2. PREMIO AL MÉRITO CIENTÍFICO TÉCNICO 2012 en la categoría: Al resultado que refleje el avance científico de mayor trascendencia y originalidad.
3. PREMIO CITMA PROVINCIAL. Contribuciones al aprendizaje automatizado a través de la Teoría de los Conjuntos Aproximados Extendida. 2012.
4. PREMIO AL MÉRITO CIENTÍFICO TÉCNICO EN LA CATEGORÍA: al resultado que refleje el avance científico de mayor trascendencia y originalidad durante el año 2011.

Trabajo futuro

A continuación se describen algunas líneas de trabajo futuras surgidas a partir de las contribuciones que han sido introducidas en esta memoria.

1. Integrar el algoritmo IFROWANN con algoritmos multiclasificadores, donde éste pueda ser combinado con técnicas al nivel de los datos (preprocesamiento) y de esta forma mejorar el comportamiento de la clasificación. En este sentido se podrían aplicar tres ideas fundamentales:
   a) La primera idea sería formar varios subconjuntos Tra a partir del conjunto original, donde en cada Tra_i se mantenga la clase minoritaria con un subconjunto de la clase mayoritaria, aplicar el clasificador IFROWANN a cada uno de éstos, para asignar un objeto a una clase cada uno de los clasificadores aporta un voto. Esta idea puede ser probada también aplicando sobre cada Tra_i un algoritmo de remuestreo para eliminar el desbalance y aplicar sobre los conjuntos resultantes el algoritmo IFROWANN con una estrategia en la que los pesos no decrezcan rápidamente, como pudiera ser $\mathscr{W}_1$.
   b) La segunda idea sería entrenar varios clasificadores (C_i) usando IFROWANN sobre el conjunto original, para cada clasificador se utilizaría una estrategia de ponderación diferente que pudieran ser $\mathscr{W}_4$, $\mathscr{W}_5$y $\mathscr{W}_6$ que fueron las que mejores resultados alcanzaron combinadas con las relaciones de inseparabilidad que mejores resultados ofrecieron: Lukasiewicz y Average. Otro criterio para seleccionar la estrategia de ponderación y la relación de inseparabilidad a utilizar pudiera ser el IR del conjunto. Antes de aplicar cada una de las variantes de IFROWANN pudiera aplicarse además un algoritmo de preprocesamiento.
   c) La tercera idea sería similar a la anterior solo que el grado de pertenencia de cada ejemplo a clase estaría determinado por la media de los resultados que se obtuvieron con cada uno de los diferentes clasificadores (cada uno usando una estrategia de ponderación y una relación de inseparabilidad diferente).
2. Mejorar la propuesta IFROWANN para que la conformación de la agregación con el operador OWA y la relación de separabilidad se seleccionen de manera automática

a partir de los datos. Como se demostró en el estudio experimental realizado en el Capítulo 4 no existe una estrategia de ponderación que obtenga los mejores resultados para cualquier índice de desbalance, como tampoco existe la mejor relación de inseparabilidad, de hecho, para cada rango de IR existe una estrategia de ponderación y una relación de inseparabilidad que se comporta mejor que el resto. Por esta razón pensamos que tanto la selección de la estrategia de ponderación como la relación de inseparabilidad pueden seleccionarse de manera automática a partir de los datos; en este sentido pudieran considerarse dos ideas fundamentales:

a) La primera idea sería basándonos en el estudio experimental realizado en el Capítulo 4 establecer un grupo de reglas que a partir del IR del conjunto de datos decida cuál es la mejor estrategia de ponderación y cuál la mejor relación de inseparabilidad.

b) La segunda sería realizar un método de envoltura que pruebe sobre el conjunto de entrenamiento las estrategias de ponderación y las relaciones de inseparabilidad y seleccione la mejor resultados ofrezca.

3. Aplicar los algoritmos diseñados a problemas reales de la ingeniería, la medicina o la educación. En este sentido ya hemos comenzado a trabajar, específicamente en área de la educación.

   La deserción escolar es un fenómeno que actualmente afecta a la mayoría de las universidades del mundo, un elevado por ciento de estudiantes abandona los estudios por disímiles razones, que pueden ser desde factores económicos o sociales hasta situaciones familiares. Determinar estas causas ha sido motivo de muchos estudios de psicólogos y pedagogos, todos con el objetivo de disminuir los crecientes índices de deserción escolar. Contando con un estudio previo donde se midieron 17 variables que pudieran estar incidiendo en la deserción escolar, se tomaron los datos de 292 estudiantes de los cuáles 75 o causaron baja o no promovieron de año. Sobre estos datos decidimos aplicar un nuevo algoritmo de clasificación probabilístico basado en la TCA para determinar con qué probabilidad puede, un estudiante que matricule en la universidad, desertar durante su primer año de estudios. Los resultados que hemos logrado son bastante alentadores pues se logrado buenos índices de acierto.
4. Extender los algoritmos diseñados para problemas de clasificación desbalanceada con más de 2 clases.

En esta tesis hemos abordado el problema del desbalance para problemas de dos clases, sin embargo en la vida real pueden aparecer problemas desbalanceados donde existan más de dos clases. Estos problemas tienen además de la agravante de la baja representatividad de uno de los conceptos el hecho de que las fronteras entre las clases pueden estar superpuestas, afectando del rendimiento de los clasificadores.
Una estrategia para abordar esta problematica es la conocida como binarización, la cual consiste en transformar el problema original en varios sub-problemas de dos clases. Existen 2 propuestas de binarización que son muy conocidas y usadas: uno-contra-uno y uno-contra-todos.
La propuesta uno-contra-uno consiste en entrenar un clasificador para todas las posibles parejas de las *n* clases. La propuesta uno-contra-todos consiste en entrenar un único clasificador para cada una de las clases del problema, considerando las instancias de la clase actual como positivos y todos los restantes como negativos.
En este sentido pudieran aplicarse dos ideas fundamentales que se describen a continuación.

a) La primera idea sería: usando previamente las mencionadas técnicas de binarización aplicar sobre cada conjunto resultante (conjunto de dos clases) los métodos de preprocesamiento híbridos propuestos en esta tesis en los Capítulos 2 y 3.
b) Aplicar sobre cada conjunto de dos clases resultante de la binarización, el algoritmo de clasificación IFROWANN y usar la estrategia de ponderación y la relación de inseparabilidad que mejor funcione (atendiendo al IR del conjunto) según las conclusiones obtenidas del estudio experimental realizado en esta tesis.

REFERENCIAS BIBLIOGRÁFICAS

J. Alcalá, L. Sánchez, S. García, M.J. del Jesús, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, and F. Herrera. KEEL: A software tool to assess evolutionary algorithms to data mining problems. *Soft Computing*, 13:3: 307–318, 2009.

J. Alcalá, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera. KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing*, 17:255–287, 2010.

ALSTOM. Instrucciones de servicio no. 188 b. interruptor de potencia de sf6 gl 312. alemania. 2002.

A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.

R. Barandela, J.S. Sánchez, V. Garcíaa, and E. Rangel. Strategies for learning in class imbalance problems. *Pattern Recognition*, 36:849–851, 2003.

G. E. A. P. A. Batista, R. C. Prati, and M.C. Monard. A study of the behaviour of several methods for balancing machine learning training data. *SIGKDD Explorations*, 6(1):20–29, 2004.

R. Bello, R. Falcon, W. Pedrycz, and J. Kacprzyk. *Granular computing: at the junction of rough sets and fuzzy sets*. Springer, 2008.

M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

A. P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159, 1997.

C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 3644:475–482, 2009.

ANSI C37.06. Ac high-voltage circuit breakers rated on a symmetrical current basis-preferred ratings and related required capabilities. *USA: American National Standards Institute, Inc*, 2000.

Y. Caballero. *Aplicación de la Teoría de los Conjuntos Aproximados en el Prepocesamiento de los Conjuntos de Entrenamiento para los Algoritmos de Aprendizaje Automatizado*. PhD thesis, Universidad Central Martha Abreu de Las Villas, 2007.

N.V. Chawla, K.W. Bowyer, L.O. Hall, and W.P. Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligent Research*, 16:321–357, 2002.

N.V. Chawla, N. Japkowicz, and A. Kolcz. Special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 6, 2004a.

N.V. Chawla, N. Japkowicz, and A. Kolcz. Editorial: special issue on learning from imbalanced data sets. *SIGKDD Explorations*, 6(1):1–6, 2004b.

N.V. Chawla, D.A. Cieslak, L.O. Hall, and A. Joshi. Automatically countering imbalance and its empirical relationship to cost. *Data Mining and Knowledge Discovery*, pages 225–252, 2008.

V. Cherkassky and F. Mulier. *Learning from data. Concepts, Theory, and Methods. Second Edition*. John Wiley & Sons, Inc., Hoboken, New Jersey, 2007.

S.K Choubey. A comparison of feature selection algorithms in the context of rough classifiers. In *Fifth IEEE International Conference on Fuzzy Systems*, 1996.

A. Chouchoulas and Q. Shen. A rough set-based approach to text classification. *Lectures Notes in Artificial Intelligence*, 11:118–127, 1999.

G. Cohen, M. Hilario, H. Sax, S. Hogonnet, and A. Geissbuhler. Learning from imbalanced data in surveillance of nosocomial infection. *Articial Intelligence in Medicine*, pages 7–18, 2006.

C. Cornelis, N. Verbiest, and R. Jensen. Ordered weighted average based fuzzy rough sets. In *Proceedings of the 5th International Conference on Rough Sets and Knowledge Technology*, pages 78–85, 2010.

T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

J. Demsâr. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

J.S. Deogun. Exploiting upper approximations in the rough set methodology. In *First International Conference on Knowledge Discovery and Data Mining*, 1995.

P. Domingos. MetaCost: a general method for making classifiers cost sensitive. In *Proceedings of Fifth International Conference on Knowledge Discovery and Data Mining (KDD99)*, pages 155–164, 1999.

T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27:861–874, 2006.

M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, 42 (4):463–484, 2012.

M. Galar, A. Fernández, E. Barrenechea, and F. Herrera. EUSBoost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling. *Pattern Recognition*, 46:3460–3471, 2013.

S. García and F. Herrera. An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2008.

S. García and F. Herrera. Evolutionary undersampling for classification with imbalanced datasets: proposals and taxonomy. *Evolutionary Computation*, 17:275–306, 2009.

S. García, A. Fernández, J. Luengo, and F. Herrera. A study of statistical techniques and performance measures for genetics–based machine learning: Accuracy and interpretability. *Soft Computing*, 13(10):959–977, 2009.

S. García, A. Fernández, J. Luengo, and F. Herrera. 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, 2010.

J. W Grzymala-Busse. Managing uncertainty in machine learning from examples. In *Proceedings of the Workshop Intelligent Information System III*, 1994.

H. Han, W.Y. Wang, and B.H. Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. *Springer-Verlag*, pages 878–887, 2005.

J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2000.

H. He and E.A. García. Learning from imbalanced data. *IEEE Transactions On Knowledge And Data Engineering*, 21(9):1263–1284, 2009.

S. Holm. A simple sequentially rejective multiple test procedure, scandinavian. *Journal of Statistics*, 6:65–70, 1979.

J. Huang and C. X. Ling. Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 17(3):299–310, 2005.

Y.M. Huang, C.M. Hung, and 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):720–747, 2006.

R. Iman and J. Davenport. Approximations of the critical region of the friedman statistic. *Communications in Statistics, Part A Theory Methods*, 9:571–595, 1980.

A. Janssen. Technical brochure no.83 cigrÉ final report of the second international enquiry on hv circuit-breaker failures and defects in service. *ELECTRA*, 1994.

R. Jensen and C. Cornelis. Fuzzy-rough instance selection. In *Proceedings of the 19th International Conference on Fuzzy Systems (FUZZ-IEEE 2010)*, pages 1776–1782, 2010.

R. Jensen and C. Cornelis. Fuzzy rough nearest neighbour classification and prediction. *Theoretical Computer Science*, 412(42):5871–5884, 2011.

J.Han. *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA., 2005.

J.H.Zhao, X.Li, and Z.Y.Dong. Online rare events detection. In *PAKDD0'7, Springer-Verlag, Berlin, Heidelberg*, 2007.

GH. John and P. Langley. Estimating continuous distributions in bayesian classifiers. In *11th conference on uncertainty in artificial intelligence*, pages 338–345. Morgan Kaufmann, San Mateo, 1995.

N.K. Kasabov. *Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering*. Massachusetts Institute of Technology, 1998.

W. Khreich, E. Granger, A. Miri, and R. Sabourin. Iterative boolean combination of classifiers in the ROC space: An application to anomaly detection with HMMs. *Pattern Recognition*, 43:2732–2752, 2010.

Y.H. Lee, P.J.H. Hu, T.H. Cheng, T.C. Huang, and W.Y. Chuang. A preclustering-based ensemble learning technique for acute appendicitis diagnoses. *Artificial Intelligence in Medicine*, 58(2):115–124, 2013.

T. Lindquist, L. Bertling, and R. Eriksson. Circuit breaker failure data and reliability modelling. *IET Generation, Transmission and Distribution*, pages 813–820, 2008.

L.M.Taft, R.S.Evans, C.R.Shyu, M.J.Egger, N.Chawla, J.A.Mitchell, S.N. Thornton, B.Bray, and M.Varner. Countering imbalanced datasets to improve adverse drug event predictive models in labor and delivery. *Journal of Biomedical Informatics*, 42:356–364, 2009.

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

M.A. Mazurowski, P.A. Habas, J.M. Zurada, J.Y. Lo, J.A. Baker, and G.D. Tourassi. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural Networks*, 21(2-3):427–436, 2008.

M.A. Mazurowskia, P. A. Habasa, J.M. Zuradaa, J.Y. Lob, J.A. Bakerb, and G.D. Tourassib. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural Networks*, 21:427–436, 2008.

G. Mazza and R. Michaca. The first international enquiry on high voltage circuit breaker failures and defects in services. *Electra*, 79:21–29, 1981.

W.S McCulloch and W. Pitts. A logical calculus od the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

L. Merschmann and A. Plastino. A lazy data mining approach for protein classification. *IEEE Transactions on Nanobioscience*, 6:36–42, 2007.

D. Miao and L. Hou. An application of rough sets to monk 's problems solving. In *9th International Conference in Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, 2003.

T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

S. Mitra and T. Acharya. *Data Mining, Multimedia, Soft Computing, and Bioinformatics*. John Wiley & Sons, INC, 2003.

M. J. Moshkov. Time complexity of decision trees. *Transactions on Rough Sets III. Lecture Notes in Computer Science*, 3400, 2005.

P.M Murphy and D.W Aha. Uci machine learning repository. *Machine Learning Databases*, 1994.

K. Napierala, J. Stefanowski, and S. Wilk. Learning from imbalanced data in presence of noisy and borderline examples. *Rough Sets and Current Trends in Computing,Lecture Notes in Computer Science*, 6086:158–167, 2010.

A. Orriols-Puig and E. Bernado-Mansilla. Evolutionary rule-based systems for imbalanced datasets. *Soft Computing*, 13(3):213–225, 2009.

Z. Pawlak. Rough sets. *International journal of Computer and Information Sciences*, 11: 145–172, 1982.

Z. Pawlak. Vagueness and uncertainty: a rough set perspective. *Computational Intelligence*, 11, 1995.

Z. Pawlak, J. Grzymala-Busse, R. Slowinski, and W. Ziarko. Rough sets. *Communications of the ACM*, 38:88–95, 1995.

X. Peng and I. King. Robust bmpm training based on second-order cone programming and its application in medical diagnosis. *Neural Networks*, 21:450–457, 2008.

V. Pitz and T. Weber. Forecasting of circuit breaker behaviour in high voltage electrical power systems: necessity for future maintenance management. *Journal of Intelligent and Robotic Systems*, pages 223–228, 2001.

J.R Quinlan. C4.5 programs for machine learning. *Morgan Kaufmann, CA*, 1988.

J.R Quinlan. C4.5 programs for machine learning. *Morgan Kaufmann, CA*, 1993.

E. Ramentol, N. Verbiest, R. Bello, Y. Caballero, Chris Cornelis, and F. Herrera. Smote-frst: A new resampling method using fuzzy rough set theory. In *Proceedings of the 10th International FLINS Conference on uncertainty Modeling in Knowledge Engineering and Decision Making (FLINS2012)*, pages 800–805, 2012.

B. Schölkopf and AJ. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. 2001.

C. Seiffert, T.Khoshgoftaar, J.VanHulse, and A.Napolitano. RUSBoost: a hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics, PartA: Systems and Humans*, 40:185–197, 2010.

D. Sheskin. Handbook of parametric and nonparametric statistical procedures, chapman & hall. *CRC*, 2003.

Z. Pawlak S.K.M., Wong S.K.M., and W. Ziarko. Rough sets: probabilistic versus deterministic approach. *International Journal of Man-Machine Studies*, 29:81–95, 1988.

A. Skowron. New directions in rough sets, data mining, and granular soft computing. In Lecture Notes in Artificial Intelligence 1711, editor, *7th International Workshop (RSFDGRC'99)*, 1999.

A. Skowron and J. Stepaniuk. Intelligent systems based on rough set approach. In *Workshop Rough Sets. State of the Art and Perspectives*, 1992.

R. Slowinski and D. Vanderpooten. Similarity relation as a basis for rough approximations. *Advances in Machine Intelligence & Soft-Computing*, 4:17–33, 1997.

Y. Sun, A. K. C. Wong, and M. S. Kamel. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(4):687–719, 2009.

P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. First Edition. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA., 2005.

S. Tan. Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications*, 28:667–671, 2005.

M. Tavallaee, N. Stakhanova, and A. Ghorbani. Toward credible evaluation of anomaly based intrusion detection methods. *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev*, 40: 516–524, 2010.

K.M. Ting. An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering*, 14 (3):659–665, 2002.

I. Tomek. Two modifications to cnn. *IEEE Transactions Systems, Man, and Communications*, pages 769–772, 1976.

S. Tsumoto. Automated extraction of hierarchical decision rules from clinical databases using rough set model. *Expert systems with Applications*, 24:189–197, 2003.

V. Vapnik. *The nature of statistical learning*. Springer, 1995.

K. Veropoulos, C. Campbell, and N. Cristianini. Controlling the sensitivity of support vector machines. In *Proceedings of the international joint conference on AI*, pages 55–60, 1999.

S. Visa and A. Ralescu. Fuzzy classifiers for imbalanced, complex classes of varying size. *Information Processing and Management of Uncertainty in Knowledge Based Systems*, pages 393–400, 2004.

L. Wang and X. Fu. *Data Mining with Computational Intelligence*. Springer Berlin Heidelberg New York, 2005.

G. Weiss and F. Provost. Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, 19:315–354, 2003.

G.M. Weiss and H. Hirsh. A quantitative study of small disjuncts. In *Proceedings of the 17th National Conf. on Artificial Inteligence*, pages 665–670, 2000.

J.C Whitaker. Ac power systems handbook. *Washington, D.C.: CRC Press LLC.*, 1999.

F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 6:80–83, 1945.

D.L Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Communications*, 3:408–421, 1972.

D.Randall Wilson and Tony R. Martinez. Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research*, 6:1–34, 1997.

I.H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques,*. Second Edition, Morgan Kaufmann Series in Data Management Systems, 2005.

X.Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. F. M. Ng, B. Liu, P. S. Yu, Z. Zhou, M. Steinbach, D.J. Hand, and D. Steinberg. Top 10 algorithms in data mining. *Knowledge And Information Systems*, 14(1):1–37, 2008.

Q. Yang and X. Wu. 10 challenging problems in data mining research. *International Journal of Information Technology and Decision Making*, 5(4):597–604, 2006.

Z. Yang, W. Tang, A. Shintemirov, and Q. Wu. Association rule mining based dissolved gas analysis for fault diagnosis of power transformers. *IEEE Trans. Syst.,Man, Cybern. C, Appl. Rev*, 39:597–610, 2009.

L. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 03)*, pages 435–442, 2003.

J. Zar. Biostatistical analysis, prentice hall. *Upper Saddle River, New Jersey*, 1999.

S. Zhang and C. Zhang. Data preparation for data mining. *Applied Artificial Intelligence*, 17(5-6):375–381, 2003.

Z.H. Zhou and X.Y. Liu. On multi-class cost-sensitive learning. *Computational Intelligence*, 26(3):232–257, 2010.

ANEXO A

Sobre el uso de pruebas no paramétricas basadas en rankings

Una prueba no paramétrica utiliza datos nominales, ordinales, u ordenados en forma de ranking. Puede resultar muy interesante la transformación de los datos de valores reales contenidos en un intervalo a datos basados en ranking, en el sentido en el que una prueba no paramétrica puede ser aplicada sobre datos típicos de una prueba paramétrica, cuando éstas no satisfacen las condiciones iniciales requeridas para el uso de esta prueba.

A continuación se explica en detalles el funcionamiento básico de cada prueba no paramétrica usada en esta investigación así como su objetivo:

- **Prueba de Friedman:** es una prueba no paramétrica equivalente una prueba de análisis de varianza ANOVA. La misma calcula el ranking de los resultados observados de un algoritmo (r_j para el algoritmo j con k algoritmos) para cada conjunto de datos, asignando al de mejor resultado ranking 1, y al peor el ranking k. Bajo la hipótesis nula, asumiendo que los resultados de los algoritmos son equivalentes y, de esta forma, sus valores de rankings también lo son, el estadígrafo de Friedman:

$$x_F^2 = \frac{12N_{ds}}{k(k+1)} \left[\sum_j R_j^2 - \frac{k(k+1)^2}{4} \right], \tag{A.1}$$

se distribuye según χ_F^2 con $k-1$ grados de libertad, siendo $R_j = \frac{1}{N_{ds}} \sum_i r_i^j$ y N_{ds} el número de conjuntos de datos. Los valores críticos para el estadígrafo de Friedman coincide con los establecidos para la distribución χ^2 cuando $N_{ds} > 10$ y $k > 5$. En el caso contrario, los valores exactos pueden ser consultados en Sheskin (2003); Visa and Ralescu (2004).

- **Prueba de Iman y Davenport:** Es una métrica derivada del estadígrafo de Friedman dado que este último se comporta de forma conservativa. El estadígrafo es:

$$F_F = \frac{(N_{ds}-1)X_F^2}{N_{ds}(k-1)-X_F^2}, \tag{A.2}$$

y se distribuye según F con $k-1$ y $(k–1)(N_{ds}–1)$ grados de libertad.

- **Prueba de Bonferroni-Dunn:** Si la hipótesis nula es rechazada en cualquiera de las pruebas anteriores se puede proceder a las pruebas post-hoc. La prueba de Bonferroni-Dunn es similar a la de Dunnet para el ANOVA y es usada para comparar un algoritmo de control contra el resto. La calidad de dos algoritmos es diferente significativamente si sus rankings medios son al menos tan grandes como sus valores críticos.

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N_{ds}}}. \tag{A.3}$$

el valor de q_α es el valor crítico de $Q^{'}$ para una comparación múltiple no paramétrica con el control(Tabla B.16 en Zar (1999)).

- **algoritmo de Holm** Holm (1979): Para el contraste al usar la prueba de Bonferroni-Dunn, se usa una prueba que chequea secuencialmente la hipótesis ordenada según su significación. Denotaremos los valores p ordenados como $p_1, p_2, ...$, en la forma $p_1 \leq p_2 \leq ... \leq p_{k-1}$. La prueba de Holm compara cada p_i con $\alpha/(k–i)$ comenzando por el valor p_i más significativo. Si p_1 es menor que $\alpha/(k–1)$, la hipótesis correspondiente es rechazada y se procede a comprar p_2 con $\alpha/(k–2)$. En el momento en el que no podamos rechazar la hipótesis p_i todas las restantes se aceptan igualmente. El estadígrafo para comparar el algoritmo i con el j es:

$$z = (R_i - R_j) / \sqrt{\frac{k(k+1)}{6N_{ds}}} \tag{A.4}$$

El valor de z se usa para encontrar el valor de la probabilidad correspondiente de una tabla de distribución normal para ser comparada con α. El algoritmo de Holm es más potente que el de Bonferroni-Dunn y no asume nada extra sobre la hipótesis probada.

- **Prueba de rangos positivos (Signed-Rank) de Wilcoxon**: Éste es la prueba no paramétrica análoga a la prueba t para muestras apareadas; por lo tanto, es una prueba dos a dos que es capaz de detectar diferencias significativas entre algoritmos. Sea d_i la diferencia entre la medida de comportamiento de dos algoritmos en la base i de un total de N_{ds} bases. Las diferencias son ranqueadas acorde a sus valores absolutos; en caso de empates se asigna el rango medio. Sea $R+$ la suma de los rangos para los datasets en los que el segundo algoritmo mejora al primero y $R-$ la suma de los rangos para el caso contrario. Los rangos $d_i = 0$ para los empates se dividen de forma homogénea entre $R+$ y $R-$ ignorando a uno si la cantidad es impar:

$$R^+ = \sum_{d_i>0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i) \tag{A.5}$$

$$R^- = \sum_{d_i<0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i) \tag{A.6}$$

Sea T el mínimo de las sumas $T = min(R^+, R^-)$. Si T es menor o igual que el valor de la distribución de Wilcoxon con N_{ds} grados de libertad (Tabla B.12 en Zar (1999)), la hipótesis nula de igualdad de medias es rechazada.

ANEXO B

Acerca del árbol de decisión C4.5

El árbol de decisión C4.5 propuesto por Quinlan en 1993 Quinlan (1993), es un clasificador que ha mostrado ser muy apropiado en el contexto de los no balanceados, es por ello que lo hemos seleccionado para los experimentos que se realizan en esta investigación.

En general la inducción de los árboles de decisión está basada en algoritmos de aprendizaje discriminativo por medio de particionamientos recursivos. Los árboles de decisión son una de las formas más sencillas de representación del conocimiento adquirido Moshkov (2005). El algoritmo C4.5, al igual que C5.0 son extensiones de algoritmo ID3, el cual solo permite el tratamiento de datos simbólicos .

C4.5 ha sido uno de los sistemas clasificadores más referenciados en la literatura, principalmente debido a su extremada robustez en un gran número de dominios (desbalance de clases por ejemplo) y su bajo costo computacional. Este algoritmo trata eficazmente los valores desconocidos calculando la ganancia de información para los valores presentes y maneja los atributos continuos. Para un conjunto de datos con m objetos y n atributos, el costo medio de construcción del árbol es de $\theta\left(mn\log_2 m\right)$, mientras que la complejidad del proceso de poda es de $\theta\left(m\left(\log_2 m\right)\right)^2$Caballero (2007).

El criterio que C4.5 usa por defecto para dividir un nodo es la entropía (gain ratio), una medida de la teoría de la información que toma distintos valores (y distintas probabilidades)

en consideración. Sea C el número de clases y $p(D,j)$ la proporción de los casos en D que pertenecen a la clase $j-ésima$. La incertidumbre residual con la que pertenece un caso del conjunto D a una determinada clase se puede expresar como:

$$Info(D) = -\sum_{j=1}^{C} p(D,j) \times \log_2 (p(D,j)) \tag{B.1}$$

Y la ganancia de información correspondiente a una prueba T con k posibles repuestas como:

$$Gain(D,T) = Info(D) - \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times Info(D) \tag{B.2}$$

La información ganada por una prueba está afectada fuertemente por el número de posibles respuestas y es máxima cuando existe un caso en cada subconjunto D_i. Por otra parte, la información que potencialmente pudiera obtenerse al dividir un conjunto en subconjuntos se basa en el conocimiento del subconjunto D_i al que pertenecerá cada caso. Esta información dividida

$$Split(D,T) = -\sum_{i=1}^{k} \frac{|D_i|}{|D|} \times \log_2 \left(\frac{|D_i|}{|D|} \right) \tag{B.3}$$

tiende a incrementarse con el número de posibles respuestas de la prueba. La entropía evalúa la pertinencia de la prueba como el índice de su ganancia de información con respecto a la división de información. La entropía de cada posible prueba se determina y se selecciona la división con mayor entropía de entre aquellas que tengan ganancia de información superior a la media.

ANEXO C

Vistas del Software “Sistema de gestión del mantenimiento de interruptores (SIGEMI)”

En la sección 3.4 se introdujo la aplicación del algoritmo SMOTE-TCAD-2T en la solución del problema del mantenimiento de los interruptores de alta potencia. Como resultado de dicha aplicación, se ha realizado un software que se encuentra en explotación en varias subestaciones eléctricas del país. SIGEMI capta los datos de varios parámetros sobre el interruptor y a partir de ellos determina si es necessario o no que el interruptor reciba mantenimiento. SIGEMI se ha convertido en una importante herramienta de apoyo a la toma de decisiones de las entidades elétricas.

A continuación se muestran algunas pantallas del sistema.

Figura C.1: Ventana Principal

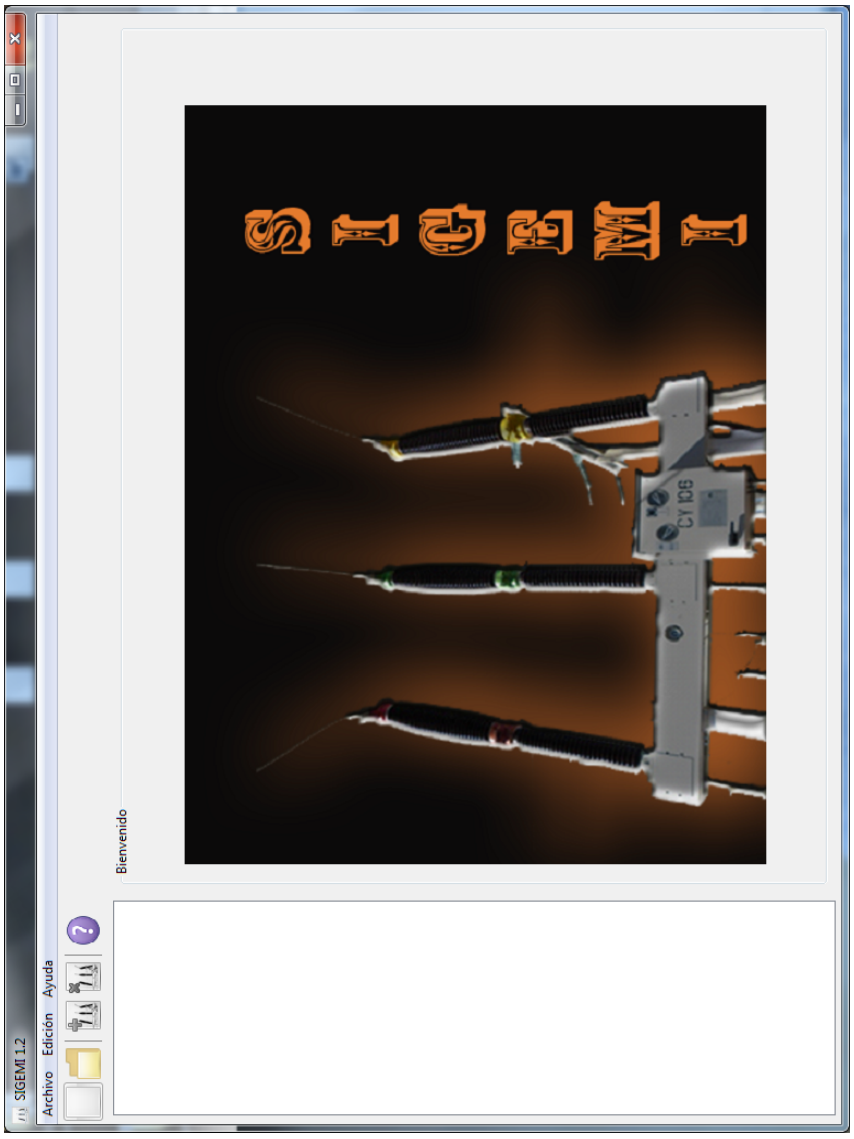


Figura C.2: Ventana Principal

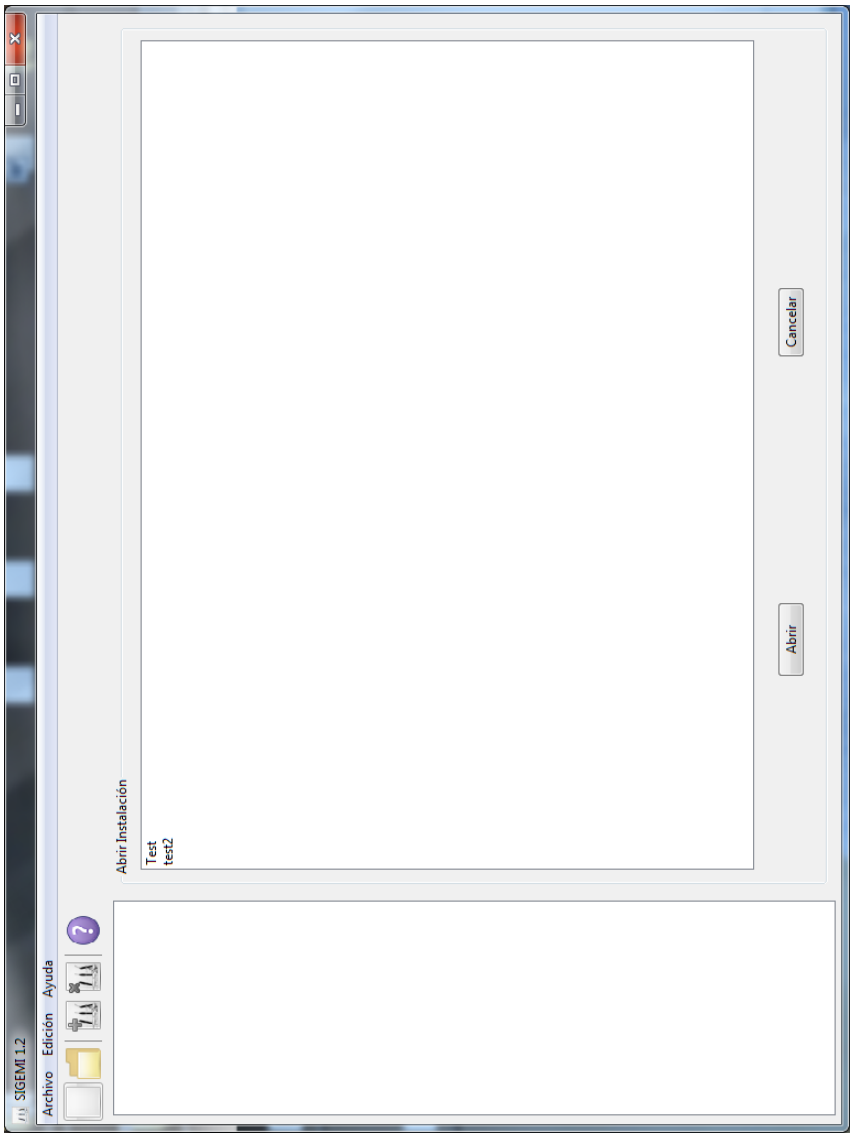


Figura C.3: Interruptor con necesidad de mantenimiento

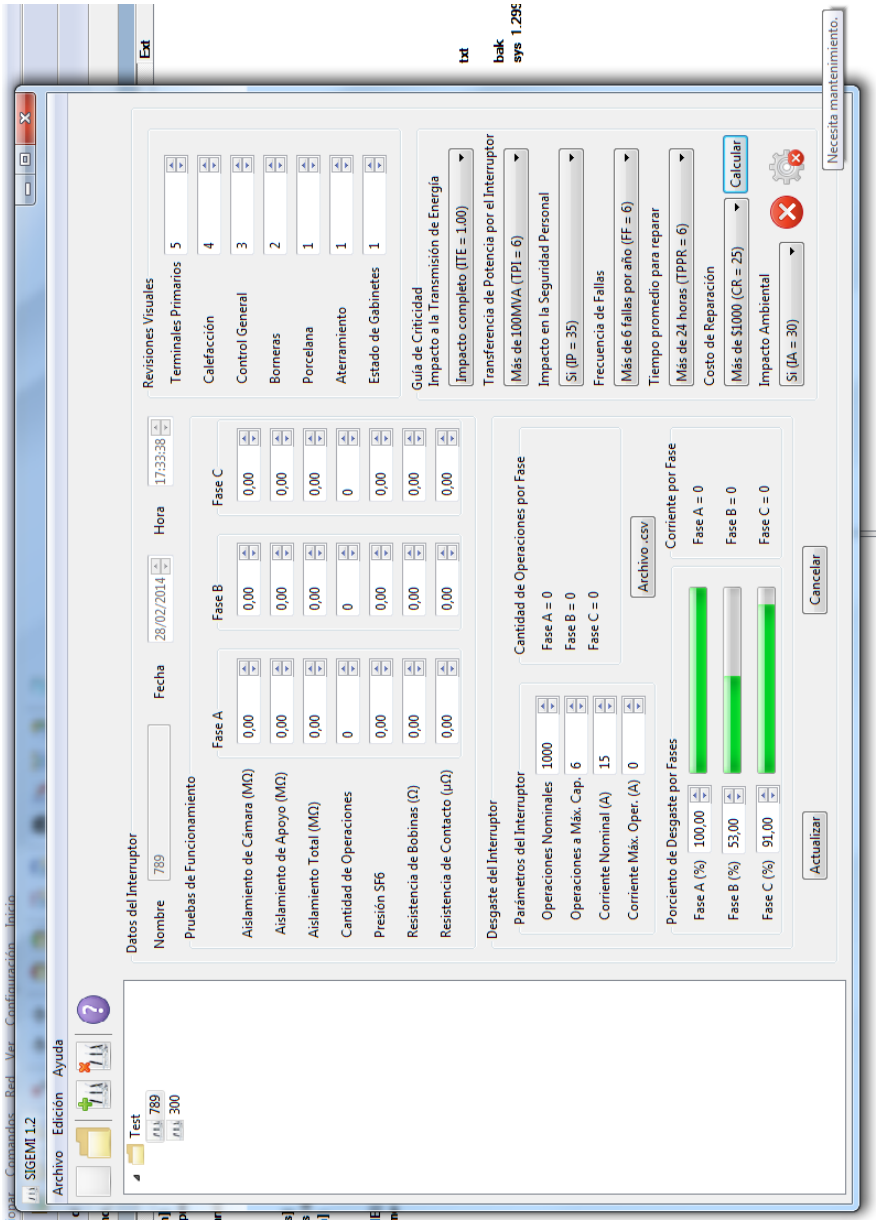


Figura C.4: Interruptor sin necesidad de mantenimiento

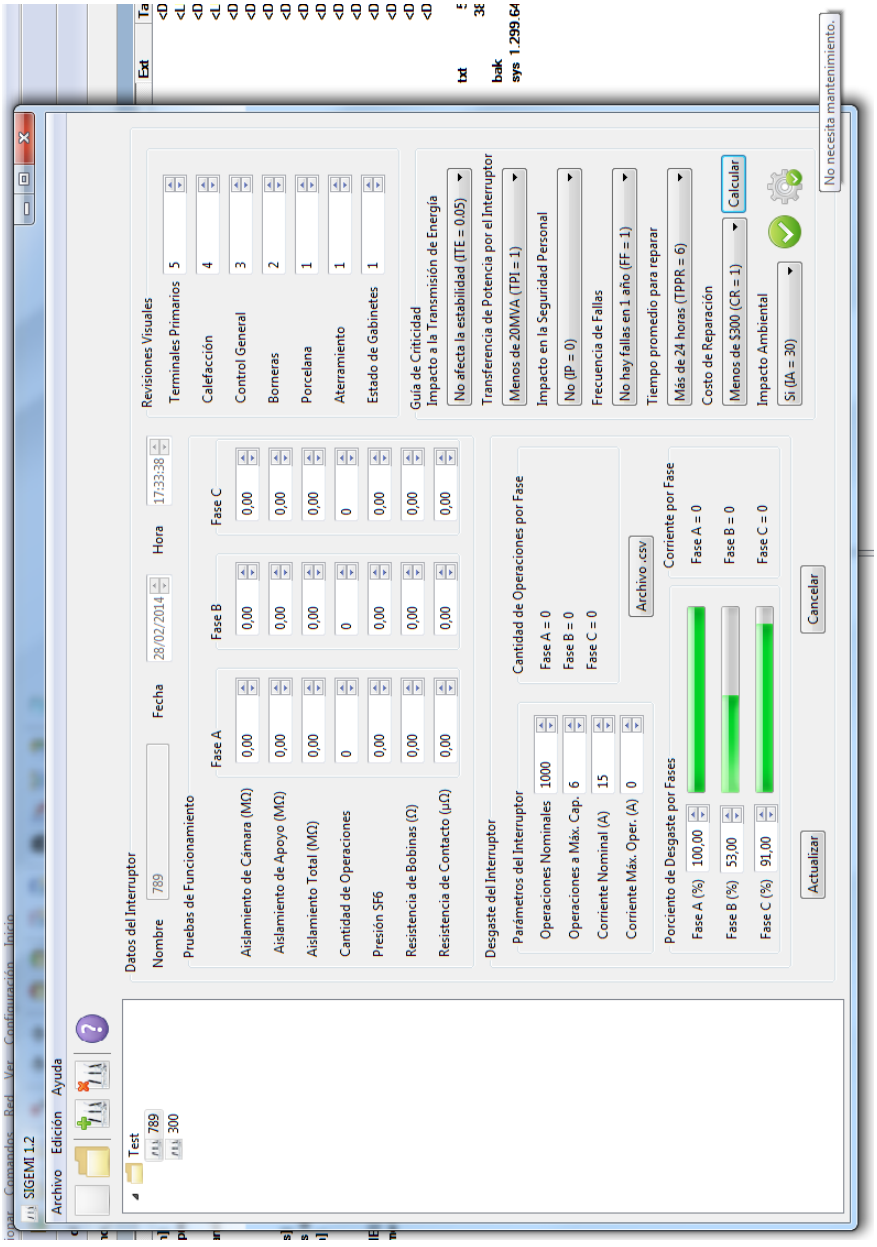


Figura C.5: Categoría de criticidad: bajo

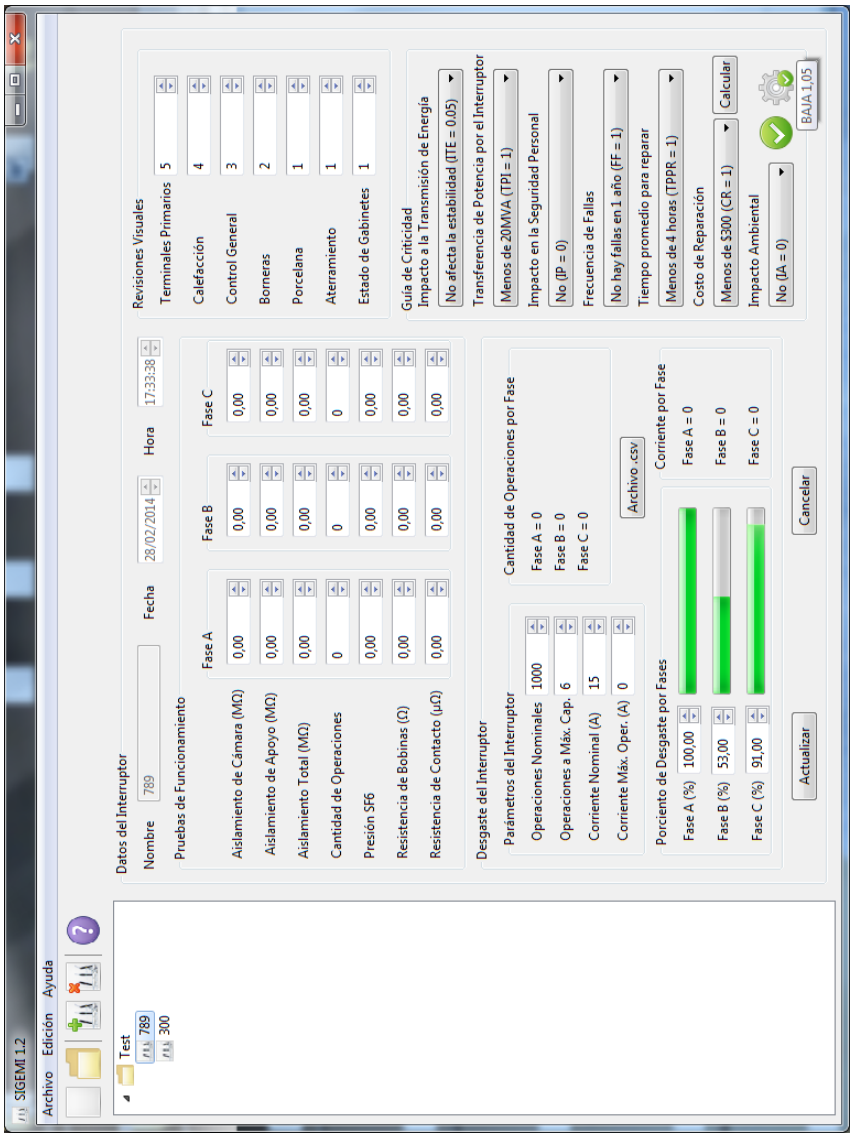


Figura C.6: Categoría de criticidad: medio

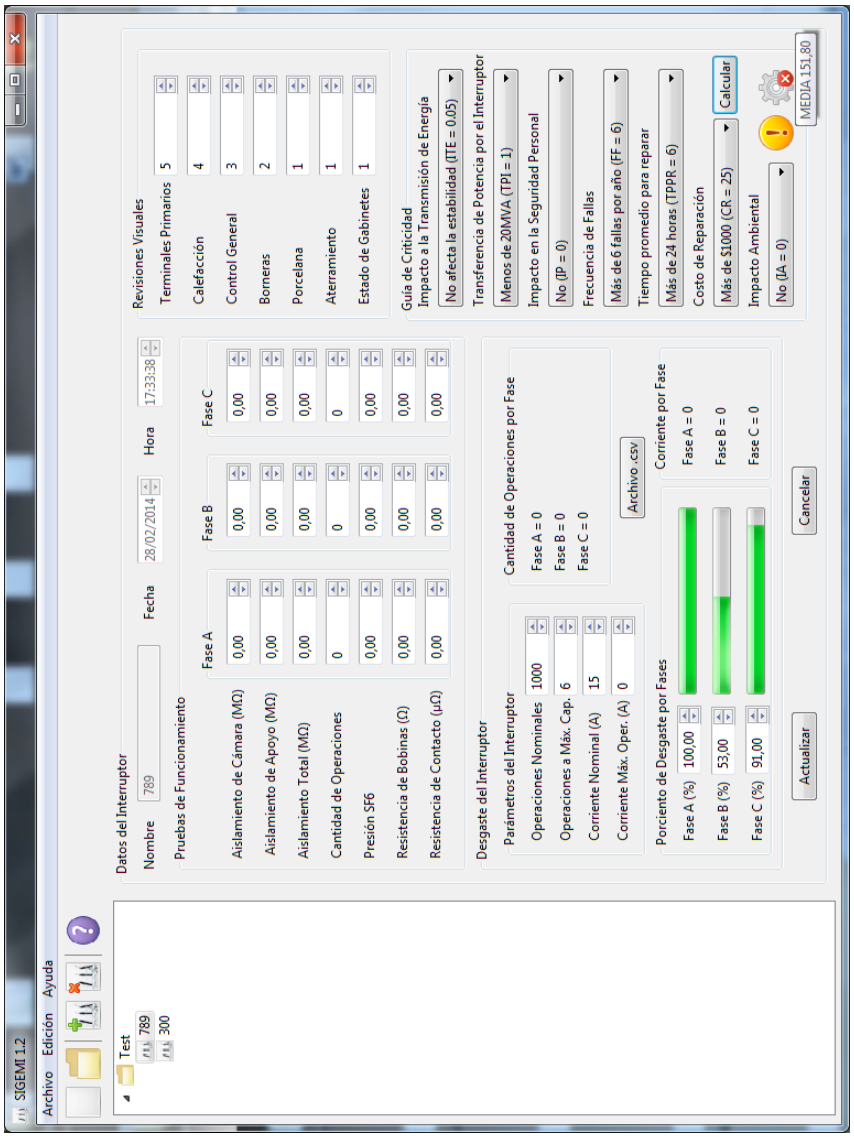


Figura C.7: Categoría de criticidad: alto

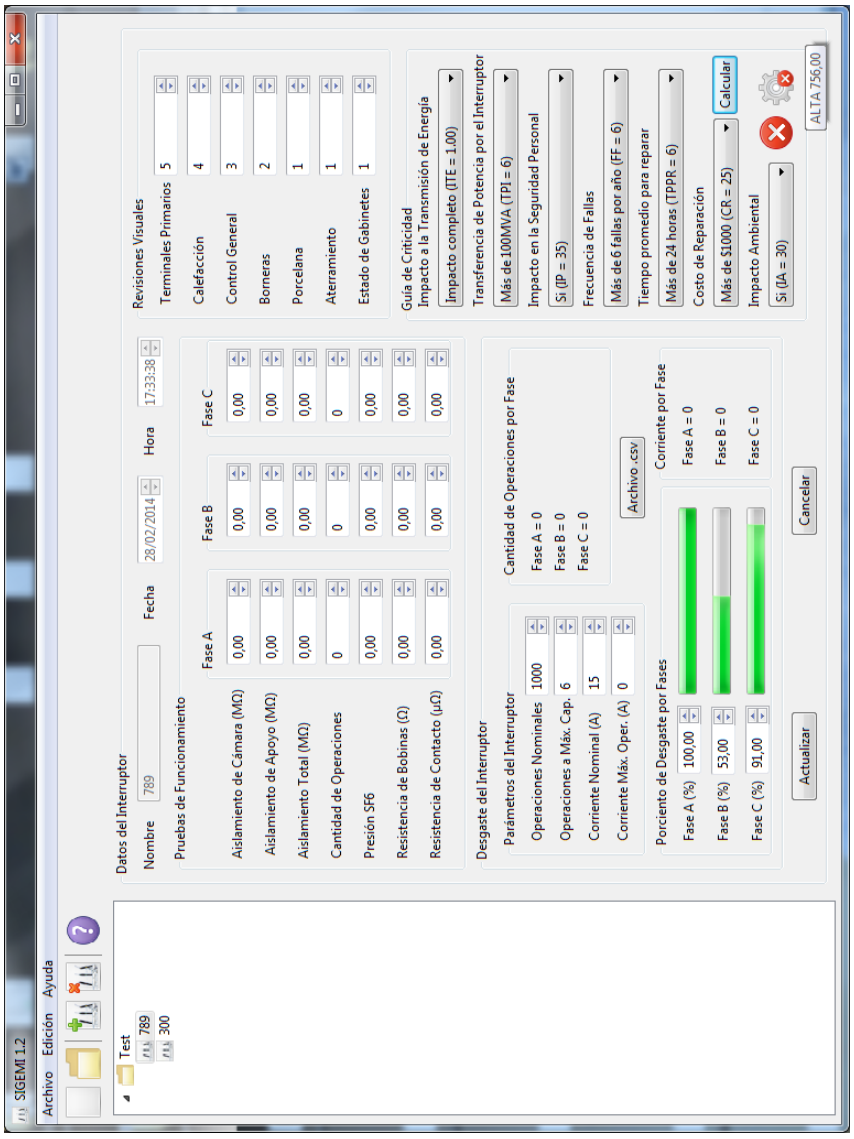


Figura C.8: Editar switch

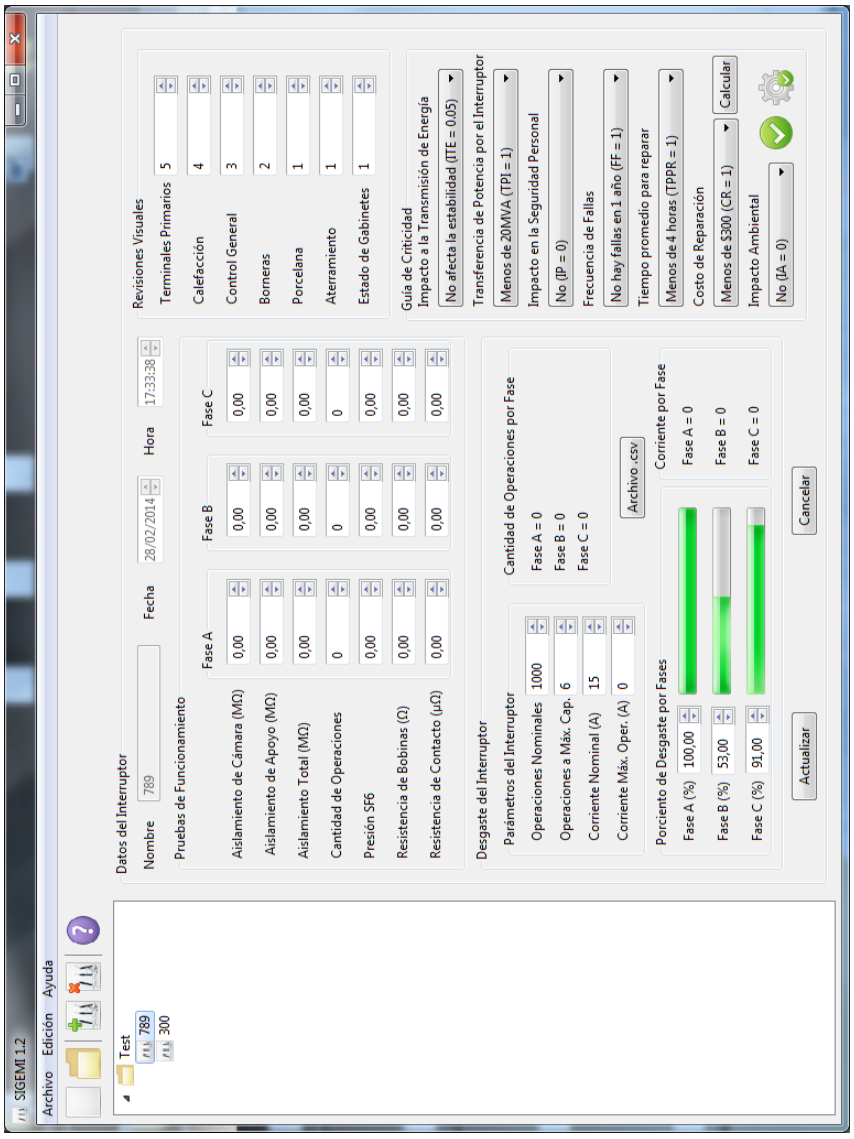
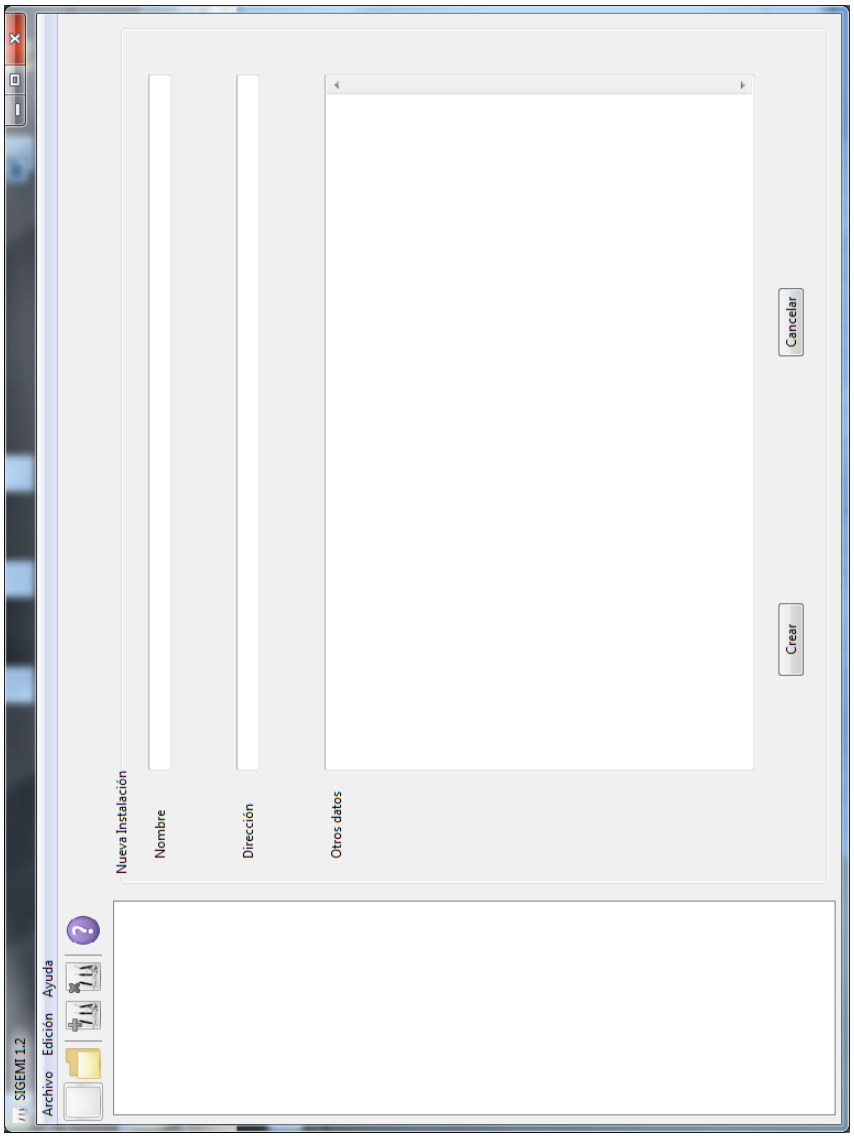


Figura C.9: Nueva Instalación



ANEXO D

AUC obtenido por los algoritmos IFROWANN

En este apéndice se presentan los resultados de AUC obtenidos por los algoritmos basados en IFROWANN introducidos en el Capítulo 4. Se muestran los resultados usando las 6 estrategias de construcción de vectores de ponderación para cada una de las 3 relaciones de separabilidad utilizadas en esta tesis: Lukasiewicz (Tabla D.1), Average (Tabla D.2) y Minimum (Tabla D.3).

Tabla D.1: Resultados de AUC para los algoritmos IFROWANN usando Lukasiewicz

Datasets	**VCCAD**	$\mathscr{W}_1$	$\mathscr{W}_2$	$\mathscr{W}_3$	$\mathscr{W}_4$	$\mathscr{W}_5$	$\mathscr{W}_6$
abalone-17_vs_7-8-9-10	0.8663	0.8187	0.8593	0.7845	0.8753	0.8692	0.9173
abalone-19_vs_10-11-12-13	0.6515	0.6852	0.5830	0.6455	0.6827	0.6828	0.7156
abalone-20_vs_8-9-10	0.8939	0.8546	0.8780	0.8155	0.9022	0.8932	0.9481
abalone-21_vs_8	0.9256	0.8992	0.8972	0.9313	0.9354	0.9204	0.9134
abalone-3_vs_11	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
abalone19	0.7241	0.7676	0.7577	0.7130	0.7401	0.7403	0.7850
abalone9-18	0.8309	0.7936	0.7891	0.8212	0.8604	0.8453	0.8993
appendicitisImb	0.7541	0.8471	0.8318	0.8653	0.8506	0.8382	0.8565
cleveland-0_vs_4	0.7806	0.7806	0.7806	0.7806	0.7806	0.7806	0.7806
cleveland-4	0.6093	0.5970	0.5917	0.6269	0.6240	0.6164	0.6052
ecoli-0-1-3-7_vs_2-6	0.8176	0.8539	0.8521	0.8215	0.8087	0.8086	0.8049
ecoli-0-1-4-6_vs_5	0.9933	0.9952	0.9923	0.9923	0.9981	0.9962	0.9962
ecoli-0-1-4-7_vs_2-3-5-6	0.9370	0.9425	0.9170	0.9437	0.9483	0.9460	0.9441
ecoli-0-1-4-7_vs_5-6	0.9625	0.9694	0.9662	0.9728	0.9703	0.9805	0.9760
ecoli-0-1_vs_2-3-5	0.9452	0.9625	0.9393	0.9441	0.9486	0.9536	0.9518
ecoli-0-1_vs_5	0.9920	0.9898	0.9852	0.9920	0.9932	0.9955	0.9920

ecoli-0-2-3-4_vs_5	0.9795	0.9877	0.9849	0.9822	0.9851	0.9864	0.9890
ecoli-0-2-6-7_vs_3-5	0.9561	0.9507	0.9351	0.9600	0.9645	0.9768	0.9683
ecoli-0-3-4-6_vs_5	0.9932	0.9919	0.9932	0.9878	0.9919	0.9878	0.9932
ecoli-0-3-4-7_vs_5-6	0.9598	0.9685	0.9556	0.9588	0.9650	0.9708	0.9762
ecoli-0-3-4_vs_5	0.9861	0.9903	0.9889	0.9903	0.9889	0.9903	0.9903
ecoli-0-4-6_vs_5	0.9959	0.9904	0.9919	0.9959	0.9972	0.9959	0.9918
ecoli-0-6-7_vs_3-5	0.9700	0.9703	0.9582	0.9840	0.9820	0.9890	0.9820
ecoli-0-6-7_vs_5	0.9575	0.9600	0.9525	0.9725	0.9700	0.9775	0.9738
ecoli-0_vs_1	0.9899	0.9926	0.9903	0.9899	0.9899	0.9931	0.9908
ecoli1	0.9278	0.9429	0.9414	0.9571	0.9566	0.9503	0.9550
ecoli2	0.9568	0.9489	0.9592	0.9507	0.9584	0.9508	0.9621
ecoli3	0.9340	0.9544	0.9472	0.9492	0.9440	0.9629	0.9534
ecoli4	0.9897	0.9865	0.9833	0.9960	0.9937	0.9937	0.9921
glass-0-1-2-3_vs_4-5-6	0.9804	0.9752	0.9740	0.9838	0.9871	0.9753	0.9813
glass-0-1-4-6_vs_2	0.7732	0.7789	0.8152	0.6673	0.7164	0.6194	0.8197
glass-0-1-5_vs_2	0.6973	0.7559	0.7355	0.6898	0.6817	0.6651	0.7720
glass-0-1-6_vs_2	0.7100	0.7067	0.7886	0.5833	0.6590	0.6314	0.8195
glass-0-1-6_vs_5	0.9686	0.9857	0.9800	0.9686	0.9771	0.9771	0.9771
glass-0-4_vs_5	0.9941	0.9941	0.9816	0.9816	0.9941	0.9941	0.9941
glass-0-6_vs_5	0.9875	0.9875	0.9825	0.9875	0.9875	0.9875	0.9875
glass0	0.8943	0.9083	0.9197	0.7676	0.9118	0.8670	0.8973
glass1	0.8738	0.8010	0.8328	0.8243	0.8822	0.8331	0.8493
glass2	0.7740	0.7418	0.8106	0.6801	0.6926	0.6036	0.7807
glass4	0.9867	0.9851	0.9827	0.9567	0.9867	0.9809	0.9801
glass5	0.9537	0.9756	0.9805	0.9585	0.9634	0.9610	0.9683
glass6	0.9836	0.9780	0.9715	0.9847	0.9836	0.9798	0.9780
habermanImb	0.5624	0.6884	0.5960	0.6150	0.5984	0.6670	0.6227
ionosphere-bredB_vs_g	0.6778	0.6778	0.6778	0.6778	0.6778	0.6778	0.6778
ionosphere-bred_vs_g	0.6867	0.6867	0.6867	0.6867	0.6867	0.6867	0.6867
iris0	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-buffer_overflow_vs_back	1,0000	0.9929	0.9869	0.9961	0.9990	0.9929	0.9974
kddcup-guess_passwd_vs_satan	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
kddcup-land_vs_portsweep	0.9994	0.9994	0.9994	0.9994	0.9994	0.9994	0.9994
kddcup-land_vs_satan	0.9994	0.9994	0.9994	0.9994	0.9994	0.9994	0.9994
kddcup-rootkit-imap_vs_back	0.9997	0.9992	0.9969	0.9989	0.9992	0.9982	0.9995
magic-hredB_vs_gredB	0.7532	0.8659	0.6356	0.8982	0.8354	0.9127	0.8914
magic-hred_vs_gred	0.7763	0.8470	0.5798	0.9228	0.8254	0.9006	0.8303
movement_libras-1	0.5769	0.5769	0.5769	0.5769	0.5769	0.5769	0.5769
new-thyroid1	1,0000	0.9984	1,0000	0.9992	1,0000	0.9952	1,0000
new-thyroid2	1,0000	0.9952	1,0000	1,0000	1,0000	0.9929	1,0000
page-blocks-1-3_vs_4	1,0000	0.9649	0.9410	0.9971	1,0000	0.9799	0.9982
page-blocks0	0.9697	0.9355	0.9569	0.9111	0.9724	0.9492	0.9701
phoneme-1redB_vs_0redB	0.7248	0.8014	0.7653	0.8210	0.7875	0.8147	0.7944
phoneme-1red_vs_0red	0.8531	0.8466	0.8340	0.8779	0.8925	0.8598	0.8696
segment-5red_vs_1-2-3	0.9822	0.9652	0.9309	0.9515	0.9791	0.9071	0.9638
segment-6red_vs_3-4-5	0.9699	0.9906	0.9808	0.9466	0.9653	0.9734	0.9897
segment-7red_vs_2-4-5-6	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
segment0	0.9999	0.9992	0.9990	0.9907	0.9998	0.9996	0.9995
shuttle-2_vs_1red	1,0000	0.9965	0.9974	0.9998	1,0000	0.9996	0.9996
shuttle-2_vs_5	1,0000	0.9995	1,0000	0.9999	1,0000	0.9991	1,0000
shuttle-6-7_vs_1redB	1,0000	0.9904	0.9922	1,0000	1,0000	1,0000	1,0000
shuttle-6_vs_2-3	0.9977	1,0000	1,0000	1,0000	0.9977	1,0000	1,0000
shuttle-c0-vs-c4	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

shuttle-c2-vs-c4	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
texture-12red_vs_13-14	0.9925	0.9925	0.9925	0.9925	0.9925	0.9925	0.9925
texture-2red_vs_3-4	0.9940	0.9897	0.9860	0.9940	0.9940	0.9938	0.9896
texture-6red_vs_7-8	0.9864	0.9864	0.9864	0.9864	0.9864	0.9864	0.9864
texture-7red_vs_2-3-4-6	0.9979	0.9979	0.9979	0.9979	0.9979	0.9979	0.9979
vehicle0	0.9749	0.9814	0.9687	0.9042	0.9798	0.9788	0.9706
vehicle1	0.7088	0.7561	0.7033	0.6660	0.7333	0.7435	0.7187
vehicle2	0.9879	0.9822	0.9715	0.9298	0.9889	0.9860	0.9728
vehicle3	0.7323	0.7745	0.7555	0.6701	0.7474	0.7593	0.7625
wdbc-MredB_vs_B	0.7171	0.7171	0.7171	0.7171	0.7171	0.7171	0.7171
wdbc-Mred_vs_B	0.7143	0.7143	0.7143	0.7143	0.7143	0.7143	0.7143
winequality-red-3_vs_5	0.7950	0.7774	0.7275	0.8479	0.8420	0.7927	0.8001
winequality-red-4	0.7713	0.7354	0.6714	0.7782	0.8024	0.6680	0.7440
winequality-red-8_vs_6	0.8648	0.8210	0.7716	0.8638	0.8688	0.8303	0.8647
winequality-red-8_vs_6-7	0.7502	0.7119	0.6905	0.7317	0.7576	0.7228	0.7754
winequality-white-3-9_vs_5	0.7978	0.8189	0.5423	0.9049	0.8474	0.8533	0.8133
winequality-white-3_vs_7	0.8872	0.8213	0.5545	0.9349	0.9182	0.9003	0.8909
winequality-white-9_vs_4	0.8788	0.8424	0.8364	0.9030	0.9030	0.8667	0.8727
wisconsinImb	0.9889	0.9882	0.9810	0.9887	0.9892	0.9883	0.9846
yeast-0-2-5-6_vs_3-7-8-9	0.8131	0.7986	0.8009	0.8371	0.8309	0.8084	0.8343
yeast-0-2-5-7-9_vs_3-6-8	0.9425	0.9203	0.9588	0.9446	0.9580	0.9286	0.9679
yeast-0-3-5-9_vs_7-8	0.7849	0.8055	0.7764	0.7833	0.7987	0.7981	0.8185
yeast-0-5-6-7-9_vs_4	0.8511	0.8693	0.8722	0.8800	0.8888	0.8832	0.9008
yeast-1-2-8-9_vs_7	0.7272	0.7960	0.8020	0.7294	0.7549	0.7754	0.7887
yeast-1-4-5-8_vs_7	0.7275	0.7029	0.7418	0.6551	0.7419	0.6759	0.7468
yeast-1_vs_7	0.8027	0.8189	0.8231	0.7660	0.8244	0.7998	0.8525
yeast-2_vs_4	0.9472	0.9466	0.9664	0.9554	0.9598	0.9449	0.9790
yeast-2_vs_8	0.8376	0.8045	0.8822	0.8659	0.8591	0.8450	0.8679
yeast1	0.7387	0.7883	0.7241	0.7599	0.7725	0.7814	0.7408
yeast3	0.9379	0.9737	0.9443	0.9489	0.9507	0.9675	0.9596
yeast4	0.8773	0.8944	0.9103	0.9032	0.9024	0.8968	0.9271
yeast5	0.9879	0.9854	0.9884	0.9890	0.9908	0.9881	0.9913
yeast6	0.9252	0.9283	0.9335	0.9412	0.9433	0.9278	0.9337
Mean	0.8905	0.8943	0.8802	0.8893	0.8998	0.8928	0.9054

Tabla D.2: Resultados de AUC para los algoritmos IFROWANN usando Average

Datasets	**VCCAD**	$\mathscr{W}_1$	$\mathscr{W}_2$	$\mathscr{W}_3$	$\mathscr{W}_4$	$\mathscr{W}_5$	$\mathscr{W}_6$
abalone-17_vs_7-8-9-10	0.8664	0.8016	0.8424	0.7473	0.8751	0.8323	0.9171
abalone-19_vs_10-11-12-13	0.6515	0.6688	0.5281	0.6302	0.6824	0.6499	0.7161
abalone-20_vs_8-9-10	0.8941	0.8380	0.8648	0.7933	0.9000	0.8469	0.9477
abalone-21_vs_8	0.9268	0.8987	0.8984	0.9219	0.9295	0.9119	0.9086
abalone-3_vs_11	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
abalone19	0.7243	0.7467	0.7324	0.7066	0.7404	0.7317	0.7842
abalone9-18	0.8309	0.7920	0.7783	0.7761	0.8604	0.8290	0.8990
appendicitisImb	0.7518	0.8500	0.8341	0.8694	0.8265	0.8524	0.8418
cleveland-0_vs_4	0.9441	0.9828	0.9777	0.9735	0.9674	0.9787	0.9747
cleveland-4	0.8229	0.8756	0.8515	0.8392	0.8311	0.8761	0.8550
ecoli-0-1-3-7_vs_2-6	0.9470	0.9691	0.9635	0.9418	0.9527	0.9436	0.9472
ecoli-0-1-4-6_vs_5	0.9808	0.9933	0.9913	0.9933	0.9913	0.9962	0.9962
ecoli-0-1-4-7_vs_2-3-5-6	0.9342	0.9572	0.9561	0.9356	0.9552	0.9491	0.9604
ecoli-0-1-4-7_vs_5-6	0.9611	0.9792	0.9773	0.9747	0.9696	0.9858	0.9799

ecoli-0-1_vs_2-3-5	0.9595	0.9514	0.9800	0.9250	0.9623	0.9323	0.9730
ecoli-0-1_vs_5	0.9807	0.9932	0.9920	0.9841	0.9864	0.9943	0.9920
ecoli-0-2-3-4_vs_5	0.9668	0.9877	0.9904	0.9821	0.9835	0.9904	0.9918
ecoli-0-2-6-7_vs_3-5	0.9203	0.9276	0.9480	0.8767	0.9197	0.9276	0.9560
ecoli-0-3-4-6_vs_5	0.9905	0.9878	0.9878	0.9865	0.9892	0.9919	0.9932
ecoli-0-3-4-7_vs_5-6	0.9555	0.9718	0.9657	0.9608	0.9607	0.9804	0.9709
ecoli-0-3-4_vs_5	0.9778	0.9889	0.9889	0.9847	0.9819	0.9903	0.9889
ecoli-0-4-6_vs_5	0.9797	0.9824	0.9824	0.9892	0.9878	0.9878	0.9878
ecoli-0-6-7_vs_3-5	0.9350	0.9575	0.9708	0.9080	0.9360	0.9500	0.9710
ecoli-0-6-7_vs_5	0.9438	0.9612	0.9563	0.9613	0.9575	0.9688	0.9700
ecoli-0_vs_1	0.9899	0.9922	0.9936	0.9899	0.9899	0.9917	0.9931
ecoli1	0.9289	0.9310	0.9131	0.9647	0.9547	0.9506	0.9411
ecoli2	0.9589	0.9410	0.9383	0.9668	0.9611	0.9543	0.9569
ecoli3	0.9283	0.9439	0.9363	0.9434	0.9401	0.9576	0.9453
ecoli4	0.9762	0.9889	0.9834	0.9802	0.9786	0.9952	0.9897
glass-0-1-2-3_vs_4-5-6	0.9713	0.9760	0.9680	0.9927	0.9860	0.9790	0.9734
glass-0-1-4-6_vs_2	0.8077	0.8057	0.8316	0.7290	0.7792	0.7313	0.8360
glass-0-1-5_vs_2	0.7328	0.7419	0.7441	0.6839	0.7366	0.7419	0.8167
glass-0-1-6_vs_2	0.7538	0.7429	0.7610	0.6886	0.7276	0.6805	0.8195
glass-0-1-6_vs_5	0.9800	0.9571	0.9629	0.9829	0.9857	0.9829	0.9857
glass-0-4_vs_5	0.9941	0.9625	0.9625	0.9941	0.9941	1,0000	1,0000
glass-0-6_vs_5	0.9950	0.9950	0.9950	0.9950	0.9950	1,0000	1,0000
glass0	0.9037	0.8950	0.8887	0.8663	0.9187	0.8946	0.8812
glass1	0.8718	0.7939	0.8441	0.7847	0.8789	0.8172	0.8287
glass2	0.8129	0.7900	0.8260	0.7555	0.7644	0.7231	0.8209
glass4	0.8908	0.9315	0.9282	0.8955	0.8989	0.9646	0.9670
glass5	0.9732	0.9293	0.9341	0.9902	0.9805	0.9902	0.9878
glass6	0.9559	0.9703	0.9811	0.9856	0.9892	0.9775	0.9838
habermanImb	0.5621	0.6828	0.6033	0.5988	0.5984	0.6544	0.6272
ionosphere-bredB_vs_g	0.9133	0.9089	0.9044	0.8978	0.8978	0.9111	0.9178
ionosphere-bred_vs_g	0.8956	0.8622	0.8578	0.9378	0.9267	0.9222	0.9156
iris0	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-buffer_overflow_vs_back	0.9999	0.9977	0.9987	0.9987	0.9995	0.9983	0.9992
kddcup-guess_passwd_vs_satan	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-land_vs_portsweep	1,0000	1,0000	1,0000	0.9998	0.9998	0.9998	0.9998
kddcup-land_vs_satan	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-rootkit-imap_vs_back	0.9999	0.9997	0.9996	0.9993	0.9999	0.9997	0.9997
magic-hredB_vs_gredB	0.7535	0.8881	0.8351	0.8031	0.8072	0.8786	0.8916
magic-hred_vs_gred	0.7574	0.8683	0.7910	0.8353	0.7986	0.8701	0.8467
movement_libras-1	0.9975	0.9837	0.9869	0.9858	0.9968	0.9898	0.9935
new-thyroid1	1,0000	1,0000	1,0000	0.9992	1,0000	1,0000	1,0000
new-thyroid2	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
page-blocks-1-3_vs_4	0.9987	0.9799	0.9818	1,0000	0.9987	0.9891	0.9927
page-blocks0	0.9699	0.9364	0.9585	0.7054	0.9723	0.9043	0.9720
phoneme-1redB_vs_0redB	0.7248	0.8015	0.7679	0.8065	0.7875	0.8136	0.7943
phoneme-1red_vs_0red	0.8531	0.8467	0.8344	0.8676	0.8925	0.8598	0.8696
segment-5red_vs_1-2-3	0.9828	0.8598	0.8925	0.9851	0.9866	0.9590	0.9687
segment-6red_vs_3-4-5	0.9929	0.9618	0.9635	0.9918	0.9965	0.9980	0.9985
segment-7red_vs_2-4-5-6	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
segment0	1,0000	0.9985	0.9996	0.9998	1,0000	0.9988	0.9998
shuttle-2_vs_1red	1,0000	0.9966	0.9974	1,0000	1,0000	0.9998	0.9996
shuttle-2_vs_5	1,0000	0.9999	1,0000	1,0000	1,0000	0.9998	1,0000
shuttle-6-7_vs_1redB	1,0000	0.9894	0.9922	1,0000	1,0000	1,0000	1,0000

shuttle-6_vs_2-3	0.9977	1,0000	1,0000	1,0000	0.9977	1,0000	1,0000
shuttle-c0-vs-c4	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
shuttle-c2-vs-c4	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
texture-12red_vs_13-14	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
texture-2red_vs_3-4	0.9934	0.9162	0.9615	0.9489	0.9946	0.9530	0.9954
texture-6red_vs_7-8	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
texture-7red_vs_2-3-4-6	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
vehicle0	0.9873	0.9184	0.9345	0.9377	0.9907	0.9675	0.9593
vehicle1	0.7513	0.7089	0.7223	0.7173	0.7864	0.7212	0.7511
vehicle2	0.9959	0.8329	0.9738	0.8569	0.9977	0.8355	0.9879
vehicle3	0.7646	0.7055	0.7414	0.7119	0.8005	0.7210	0.7722
wdbc-MredB_vs_B	0.9954	0.9842	0.9916	0.9898	0.9935	0.9879	0.9926
wdbc-Mred_vs_B	0.9972	0.9972	0.9972	0.9986	0.9986	0.9986	0.9986
winequality-red-3_vs_5	0.8012	0.8108	0.8284	0.7798	0.8085	0.7741	0.8204
winequality-red-4	0.7462	0.7486	0.7100	0.7562	0.7631	0.7495	0.7374
winequality-red-8_vs_6	0.8592	0.8578	0.8727	0.7780	0.8504	0.8097	0.8722
winequality-red-8_vs_6-7	0.7524	0.7778	0.8013	0.6878	0.7386	0.7087	0.7825
winequality-white-3-9_vs_5	0.6831	0.7421	0.6901	0.7344	0.7024	0.7241	0.6671
winequality-white-3_vs_7	0.8142	0.8102	0.7864	0.8563	0.8372	0.8338	0.8335
winequality-white-9_vs_4	0.9333	0.9152	0.9212	0.9333	0.9333	0.9152	0.9152
wisconsinImb	0.9942	0.9950	0.9956	0.9953	0.9959	0.9951	0.9957
yeast-0-2-5-6_vs_3-7-8-9	0.8131	0.8210	0.8322	0.7649	0.8311	0.8278	0.8344
yeast-0-2-5-7-9_vs_3-6-8	0.9426	0.9231	0.9610	0.8956	0.9583	0.9276	0.9660
yeast-0-3-5-9_vs_7-8	0.7860	0.8188	0.7796	0.7777	0.8020	0.8065	0.8197
yeast-0-5-6-7-9_vs_4	0.8517	0.8598	0.8630	0.8642	0.8844	0.8794	0.8959
yeast-1-2-8-9_vs_7	0.7275	0.8078	0.7916	0.7181	0.7562	0.8030	0.7880
yeast-1-4-5-8_vs_7	0.7290	0.7074	0.7258	0.6642	0.7449	0.7100	0.7493
yeast-1_vs_7	0.8086	0.8398	0.8482	0.7723	0.8318	0.8282	0.8536
yeast-2_vs_4	0.9470	0.9447	0.9724	0.9387	0.9618	0.9436	0.9732
yeast-2_vs_8	0.8376	0.8736	0.8833	0.8282	0.8601	0.8332	0.8700
yeast1	0.7384	0.7864	0.7069	0.7494	0.7718	0.8007	0.7352
yeast3	0.9378	0.9732	0.9295	0.9452	0.9506	0.9757	0.9593
yeast4	0.8770	0.8900	0.9095	0.8957	0.9030	0.9020	0.9203
yeast5	0.9881	0.9894	0.9895	0.9889	0.9909	0.9898	0.9914
yeast6	0.9254	0.9333	0.9263	0.9483	0.9441	0.9415	0.9347
Mean	0.9083	0.9098	0.9094	0.8990	0.9181	0.9122	0.9256

Tabla D.3: Resultados de AUC para los algoritmos IFROWANN usando Minimun

Datasets	**VCCAD**	$\mathscr{W}_1$	$\mathscr{W}_2$	$\mathscr{W}_3$	$\mathscr{W}_4$	$\mathscr{W}_5$	$\mathscr{W}_6$
abalone-17_vs_7-8-9-10	0.8354	0.8643	0.8719	0.7871	0.8629	0.8968	0.9263
abalone-19_vs_10-11-12-13	0.6839	0.7253	0.6149	0.6447	0.6987	0.6865	0.7432
abalone-20_vs_8-9-10	0.9247	0.9121	0.9093	0.8757	0.9401	0.9252	0.9580
abalone-21_vs_8	0.9142	0.9231	0.9207	0.9375	0.9523	0.9304	0.9390
abalone-3_vs_11	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
abalone19	0.7438	0.7829	0.7884	0.6999	0.7572	0.7418	0.8113
abalone9-18	0.8399	0.8680	0.8625	0.8389	0.8687	0.8895	0.9125
appendicitisImb	0.8082	0.8424	0.8000	0.8347	0.8094	0.8447	0.8088
cleveland-0_vs_4	0.9295	0.9339	0.9167	0.8801	0.9176	0.9004	0.9268
cleveland-4	0.8346	0.8189	0.8106	0.6915	0.7692	0.7189	0.7837
ecoli-0-1-3-7_vs_2-6	0.9196	0.9216	0.9379	0.8889	0.9052	0.8761	0.8960
ecoli-0-1-4-6_vs_5	0.9125	0.9212	0.9154	0.9327	0.9231	0.9385	0.9510

ecoli-0-1-4-7_vs_2-3-5-6	0.8943	0.9265	0.9076	0.9140	0.9154	0.9228	0.9275
ecoli-0-1-4-7_vs_5-6	0.9213	0.9249	0.9379	0.9288	0.9379	0.9386	0.9470
ecoli-0-1_vs_2-3-5	0.9007	0.9298	0.9200	0.9143	0.9225	0.9266	0.9252
ecoli-0-1_vs_5	0.9205	0.9284	0.9307	0.9330	0.9352	0.9307	0.9477
ecoli-0-2-3-4_vs_5	0.8835	0.9185	0.9223	0.8965	0.8948	0.9226	0.9280
ecoli-0-2-6-7_vs_3-5	0.9138	0.8934	0.9149	0.9133	0.9210	0.9105	0.9345
ecoli-0-3-4-6_vs_5	0.8959	0.9270	0.9284	0.9081	0.9068	0.9284	0.9500
ecoli-0-3-4-7_vs_5-6	0.9087	0.9249	0.9291	0.9164	0.9223	0.9265	0.9342
ecoli-0-3-4_vs_5	0.8882	0.9250	0.9306	0.9042	0.8986	0.9236	0.9375
ecoli-0-4-6_vs_5	0.8863	0.9138	0.9357	0.8908	0.9018	0.9232	0.9452
ecoli-0-6-7_vs_3-5	0.9570	0.9450	0.9505	0.9453	0.9567	0.9553	0.9650
ecoli-0-6-7_vs_5	0.8894	0.9138	0.9175	0.9225	0.9138	0.9238	0.9238
ecoli-0_vs_1	0.9885	0.9899	0.9890	0.9908	0.9917	0.9904	0.9899
ecoli1	0.9287	0.9371	0.9216	0.9448	0.9507	0.9469	0.9365
ecoli2	0.9490	0.9454	0.9456	0.9570	0.9567	0.9508	0.9567
ecoli3	0.8993	0.9439	0.9259	0.9248	0.9248	0.9586	0.9349
ecoli4	0.9429	0.9857	0.9588	0.9484	0.9476	0.9825	0.9612
glass-0-1-2-3_vs_4-5-6	0.9766	0.9706	0.9750	0.9844	0.9838	0.9771	0.9761
glass-0-1-4-6_vs_2	0.6946	0.7144	0.7374	0.6591	0.6710	0.6722	0.7050
glass-0-1-5_vs_2	0.5858	0.6978	0.7194	0.5909	0.5957	0.5753	0.6296
glass-0-1-6_vs_2	0.6076	0.6510	0.6529	0.6352	0.6110	0.6267	0.6605
glass-0-1-6_vs_5	0.9400	0.9000	0.9086	0.9571	0.9543	0.9771	0.9771
glass-0-4_vs_5	0.9816	0.9445	0.9445	0.9816	0.9820	0.9504	0.9507
glass-0-6_vs_5	1,0000	0.9597	0.9547	0.9950	0.9950	0.9950	1,0000
glass0	0.8870	0.8482	0.8361	0.8281	0.9043	0.8693	0.8031
glass1	0.8000	0.7480	0.7480	0.7402	0.8110	0.7156	0.7217
glass2	0.7160	0.7444	0.7181	0.7029	0.7075	0.6945	0.7315
glass4	0.9565	0.9115	0.9161	0.9710	0.9654	0.9503	0.9532
glass5	0.9146	0.8829	0.8976	0.9488	0.9488	0.9561	0.9756
glass6	0.8971	0.9445	0.9420	0.9468	0.8948	0.9517	0.9335
habermanImb	0.6083	0.6161	0.5368	0.6055	0.6186	0.5989	0.5865
ionosphere-bredB_vs_g	0.9844	0.9556	0.9311	0.9978	0.9978	0.9956	0.9911
ionosphere-bred_vs_g	0.9133	0.9133	0.9133	0.9667	0.9578	0.9600	0.9511
iris0	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-buffer_overflow_vs_back	0.9927	0.9914	0.9752	0.9976	0.9925	0.9937	0.9967
kddcup-guess_passwd_vs_satan	1,0000	0.9999	0.9998	0.9999	1,0000	1,0000	0.9999
kddcup-land_vs_portsweep	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-land_vs_satan	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-rootkit-imap_vs_back	0.9999	0.9991	0.9995	0.9994	0.9999	0.9985	0.9991
magic-hredB_vs_gredB	0.7222	0.8516	0.8014	0.8303	0.7853	0.8796	0.8560
magic-hred_vs_gred	0.7943	0.8379	0.7897	0.8384	0.8257	0.8840	0.8672
movement_libras-1	0.9962	0.9816	0.9883	0.9865	0.9948	0.9828	0.9898
new-thyroid1	0.9992	0.9937	0.9984	0.9810	0.9992	0.9937	0.9992
new-thyroid2	0.9984	0.9921	0.9984	0.9881	0.9992	0.9897	0.9992
page-blocks-1-3_vs_4	0.9965	0.9645	0.9224	0.9918	0.9939	0.9763	0.9775
page-blocks0	0.9511	0.8950	0.9247	0.7644	0.9566	0.8926	0.9469
phoneme-1redB_vs_0redB	0.7166	0.7660	0.7337	0.7408	0.7682	0.7584	0.7856
phoneme-1red_vs_0red	0.8397	0.8024	0.7936	0.7741	0.8693	0.8102	0.8466
segment-5red_vs_1-2-3	0.9554	0.8890	0.9133	0.9321	0.9640	0.8883	0.9394
segment-6red_vs_3-4-5	0.9875	0.9670	0.9717	0.9805	0.9887	0.9840	0.9956
segment-7red_vs_2-4-5-6	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
segment0	0.9988	0.9302	0.9868	0.9805	0.9981	0.9679	0.9938
shuttle-2_vs_1red	0.9953	0.9610	0.9751	0.9992	0.9948	0.9916	0.9923

shuttle-2_vs_5	1,0000	0.9996	1,0000	0.9998	1,0000	0.9995	1,0000
shuttle-6-7_vs_1redB	1,0000	0.9904	0.9974	1,0000	1,0000	0.9999	1,0000
shuttle-6_vs_2-3	1,0000	0.9295	0.9318	0.9977	0.9977	0.9864	0.9750
shuttle-c0-vs-c4	1,0000	0.9999	0.9999	1,0000	1,0000	1,0000	1,0000
shuttle-c2-vs-c4	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
texture-12red_vs_13-14	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
texture-2red_vs_3-4	0.9988	0.9685	0.9881	0.9949	0.9988	0.9940	0.9980
texture-6red_vs_7-8	1,0000	0.9990	1,0000	1,0000	1,0000	0.9998	1,0000
texture-7red_vs_2-3-4-6	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
vehicle0	0.9501	0.9066	0.8921	0.8861	0.9611	0.9214	0.9321
vehicle1	0.7387	0.7116	0.7370	0.6260	0.7724	0.7025	0.7632
vehicle2	0.9763	0.8265	0.8992	0.8937	0.9780	0.8487	0.9387
vehicle3	0.7865	0.6925	0.7545	0.6646	0.8130	0.6926	0.7814
wdbc-MredB_vs_B	0.9616	0.9673	0.9664	0.9786	0.9757	0.9721	0.9748
wdbc-Mred_vs_B	1,0000	0.9915	0.9944	0.9972	0.9986	0.9958	0.9958
winequality-red-3_vs_5	0.7774	0.7814	0.7977	0.7651	0.8056	0.7571	0.7940
winequality-red-4	0.6856	0.7338	0.6588	0.7243	0.7150	0.7204	0.6973
winequality-red-8_vs_6	0.8675	0.8709	0.8790	0.8376	0.8729	0.8342	0.8736
winequality-red-8_vs_6-7	0.8033	0.8077	0.8244	0.7432	0.7983	0.7472	0.8251
winequality-white-3-9_vs_5	0.7398	0.7814	0.7418	0.7353	0.7652	0.7914	0.7507
winequality-white-3_vs_7	0.8841	0.8037	0.7980	0.8852	0.9116	0.8693	0.9105
winequality-white-9_vs_4	0.9394	0.9152	0.9152	0.9394	0.9394	0.9273	0.9212
wisconsinImb	0.9801	0.9913	0.9912	0.9908	0.9916	0.9917	0.9914
yeast-0-2-5-6_vs_3-7-8-9	0.7911	0.8208	0.8164	0.6809	0.8088	0.8305	0.8239
yeast-0-2-5-7-9_vs_3-6-8	0.9323	0.9368	0.9429	0.8580	0.9514	0.9407	0.9556
yeast-0-3-5-9_vs_7-8	0.7440	0.8088	0.7688	0.6951	0.7882	0.8110	0.8041
yeast-0-5-6-7-9_vs_4	0.8419	0.8583	0.8548	0.8423	0.8786	0.8666	0.8849
yeast-1-2-8-9_vs_7	0.7070	0.7747	0.7612	0.6962	0.7164	0.7413	0.7496
yeast-1-4-5-8_vs_7	0.6815	0.6586	0.7009	0.5855	0.6780	0.6221	0.6905
yeast-1_vs_7	0.7741	0.8245	0.8213	0.6818	0.7669	0.7752	0.8096
yeast-2_vs_4	0.9202	0.9143	0.9421	0.8975	0.9411	0.9137	0.9510
yeast-2_vs_8	0.8019	0.7690	0.8111	0.8250	0.8362	0.8057	0.8477
yeast1	0.7210	0.7692	0.6680	0.7095	0.7561	0.7689	0.6958
yeast3	0.9405	0.9627	0.9105	0.9266	0.9511	0.9564	0.9463
yeast4	0.8599	0.8686	0.8820	0.8587	0.8840	0.8761	0.8988
yeast5	0.9931	0.9843	0.9867	0.9855	0.9930	0.9881	0.9927
yeast6	0.9121	0.9177	0.9092	0.9347	0.9279	0.9282	0.9191
Mean	0.8925	0.8936	0.8908	0.8813	0.9030	0.8955	0.9071

ANEXO E

AUC obtenido por los algoritmos del estado del arte seleccionados para comparar con las variantes IFROWANN

En este apéndice se presentan los resultados experimentales de los algoritmos del estado del arte selecionados para comparar con los algoritmos IFROWANN introducidos en el Capítulo 4.

En la tabla E.1 se muestran los resultados de AUC para los algoritmos de preprocesamiento seleccionados usando como clasificadores C4.5, kNN y SVM. Las columnas de las tablas se corresponden con:

- SMOTE-kNN: S-kNN
- SMOTE-C4.5: S-C4.5
- SMOTE-SVM: S-SVM
- SMOTE-RSB*-kNN: B-kNN
- SMOTE-RSB*-C4.5: B-C4.5
- SMOTE-RSB*-SVM: B-SVM
- SMOTE-ENN-kNN: E-kNN

- SMOTE-ENN-C4.5: E-C4.5
- SMOTE-ENN-SVM: E-SVM

Tabla E.1: Resultados de AUC para los algoritmos de preprocesamiento

Datasets	**S-kNN**	**S-C4.5**	**S-SVM**	**B-kNN**	**B-C4.5**	**B-SVM**	**E-kNN**	**E-C4.5**	**E-SVM**
abalone-17_vs_7-8-9-10	0.8583	0.7177	0.9277	0.8518	0.8045	0.9283	0.8595	0.7370	0.9298
abalone-19_vs_10-11-12-13	0.6400	0.5532	0.7551	0.6294	0.5936	0.7569	0.6504	0.5436	0.7610
abalone-20_vs_8-9-10	0.8996	0.7461	0.9583	0.8871	0.6970	0.9586	0.9001	0.8059	0.9625
abalone-21_vs_8	0.8418	0.8396	0.9036	0.8657	0.7982	0.9024	0.8687	0.8539	0.9021
abalone-3_vs_11	1,0000	0.9979	1,000	1,0000	0.9959	1,0000	0.9990	0.9959	0.9990
abalone19	0.7010	0.5729	0.8398	0.7193	0.5676	0.8391	0.7050	0.5401	0.8395
abalone9-18	0.8480	0.7788	0.9158	0.8555	0.8003	0.9141	0.8413	0.7850	0.9175
appendicitisImb	0.7997	0.6697	0.8576	0.7721	0.7341	0.8577	0.7509	0.7276	0.8459
cleveland-0_vs_4	0.9559	0.7951	0.9593	0.9577	0.7220	0.9707	0.9309	0.7812	0.9906
cleveland-4	0.8176	0.4361	0.8651	0.7628	0.4376	0.8797	0.8333	0.6192	0.8715
ecoli-0-1-3-7_vs_2-6	0.8680	0.6054	0.9563	0.8317	0.8427	0.9618	0.8707	0.8290	0.9545
ecoli-0-1-4-6_vs_5	0.9870	0.8327	0.9183	0.9798	0.8399	0.9202	0.9788	0.8274	0.8990
ecoli-0-1-4-7_vs_2-3-5-6	0.9422	0.8812	0.9184	0.9445	0.8791	0.9290	0.9422	0.9199	0.9269
ecoli-0-1-4-7_vs_5-6	0.9373	0.8906	0.9333	0.9673	0.8348	0.9405	0.9498	0.8096	0.9358
ecoli-0-1_vs_2-3-5	0.9205	0.8363	0.9575	0.9307	0.8290	0.9518	0.9291	0.8795	0.9561
ecoli-0-1_vs_5	0.9278	0.858	0.8909	0.9278	0.8693	0.8886	0.9273	0.8807	0.8864
ecoli-0-2-3-4_vs_5	0.9489	0.9098	0.884	0.9455	0.8808	0.8854	0.9482	0.8901	0.9061
ecoli-0-2-6-7_vs_3-5	0.9580	0.8263	0.9035	0.9249	0.8517	0.8975	0.9351	0.8045	0.8938
ecoli-0-3-4-6_vs_5	0.9568	0.9385	0.8541	0.9750	0.8730	0.8703	0.9547	0.9223	0.8649
ecoli-0-3-4-7_vs_5-6	0.9551	0.847	0.9385	0.9594	0.8762	0.9377	0.9521	0.8962	0.9317
ecoli-0-3-4_vs_5	0.9715	0.8972	0.9028	0.9764	0.9264	0.9042	0.9722	0.9090	0.9028
ecoli-0-4-6_vs_5	0.9694	0.9385	0.88	0.9775	0.8777	0.8919	0.9762	0.9294	0.8758
ecoli-0-6-7_vs_3-5	0.9385	0.882	0.849	0.9664	0.8321	0.8540	0.9425	0.8269	0.8463
ecoli-0-6-7_vs_5	0.9131	0.7994	0.8663	0.9344	0.8044	0.8663	0.9356	0.7113	0.8600
ecoli-0_vs_1	0.9846	0.9832	0.9926	0.9897	0.9742	0.9922	0.9846	0.9832	0.9906
ecoli1	0.9661	0.9241	0.9482	0.9646	0.8931	0.9485	0.9592	0.8949	0.9452
ecoli2	0.9580	0.9031	0.9365	0.9634	0.9162	0.9345	0.9541	0.8858	0.9344
ecoli3	0.9450	0.895	0.9344	0.9431	0.8734	0.9301	0.9474	0.8725	0.9348
ecoli4	0.9944	0.9338	0.9905	0.9960	0.8853	0.9913	0.9933	0.8845	0.9905
glass-0-1-2-3_vs_4-5-6	0.9735	0.8277	0.9635	0.9686	0.8736	0.9677	0.9616	0.9323	0.9609
glass-0-1-4-6_vs_2	0.7870	0.7059	0.7476	0.7469	0.7491	0.7502	0.7850	0.8100	0.7224
glass-0-1-5_vs_2	0.7328	0.7503	0.6194	0.7556	0.5790	0.6457	0.7234	0.6532	0.6376
glass-0-1-6_vs_2	0.7343	0.7362	0.6157	0.7379	0.6769	0.6410	0.7479	0.6695	0.6719
glass-0-1-6_vs_5	0.9843	0.95	0.9314	0.9729	0.9143	0.9343	0.9800	0.9129	0.9314
glass-0-4_vs_5	0.9853	0.9879	0.9504	0.9662	0.9875	0.9629	0.9794	0.9816	0.9504
glass-0-6_vs_5	0.9875	0.95	0.9542	0.9925	0.9350	0.9692	0.9900	0.9800	0.9489
glass0	0.8904	0.7965	0.8205	0.8850	0.7985	0.8150	0.8814	0.8199	0.8271
glass1	0.8392	0.6936	0.6013	0.8425	0.7810	0.5863	0.8575	0.7493	0.6193
glass2	0.8072	0.7443	0.708	0.7375	0.7868	0.6806	0.7948	0.8168	0.7063
glass4	0.9690	0.9	0.8818	0.9490	0.8633	0.8684	0.9641	0.9193	0.8722
glass5	0.9878	0.9732	0.8976	0.9866	0.9768	0.9049	0.9854	0.8890	0.9098
glass6	0.9538	0.8781	0.9317	0.9554	0.8736	0.9229	0.9567	0.9207	0.9378
habermanImb	0.6104	0.6263	0.6741	0.6025	0.6481	0.6719	0.5935	0.6425	0.6920
ionosphere-bredB_vs_g	0.9289	0.6911	0.9356	0.9300	0.6933	0.9200	0.9311	0.8778	0.9467
ionosphere-bred_vs_g	0.8411	0.6033	0.8333	0.8256	0.8411	0.8511	0.8589	0.7678	0.8400
iris0	1,0000	0.99	1,0000	1,0000	0.9900	1,0000	1,0000	0.9900	1,0000

kddcup-buffer_overflow_vs_back	0.9997	0.9833	1,0000	0.9997	0.9833	1,0000	0.9997	0.9833	1,0000
kddcup-guess_passwd_vs_satan	1,0000	0.9994	1,0000	1,0000	0.9994	1,0000	1,0000	0.9994	1,0000
kddcup-land_vs_portsweep	0.9998	1,0000	1,0000	0.9998	1,0000	1,0000	0.9998	1,0000	0.9995
kddcup-land_vs_satan	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-rootkit-imap_vs_back	0.9999	0.9995	1,0000	0.9999	0.9996	1,0000	0.9999	0.9995	1,0000
magic-hredB_vs_gredB	0.8892	0.7873	0.8933	0.8927	0.7104	0.8944	0.8806	0.7467	0.8919
magic-hred_vs_gred	0.8851	0.696	0.8518	0.8736	0.6177	0.8503	0.8892	0.7715	0.8501
movement_libras-1	0.9932	0.8244	0.9596	0.9899	0.8424	0.9578	0.9905	0.8215	0.9695
new-thyroid1	0.9952	0.9595	0.9992	0.9964	0.9575	1,0000	0.9956	0.9825	1,0000
new-thyroid2	0.9925	0.896	0.9984	0.9964	0.9508	0.9984	0.9929	0.9083	0.9984
page-blocks-1-3_vs_4	0.9867	0.9934	0.9395	0.9810	0.9944	0.9549	0.9852	0.9921	0.9408
page-blocks0	0.9430	0.9549	0.9298	0.9388	0.9585	0.9278	0.9435	0.9322	0.9256
phoneme-1redB_vs_0redB	0.7989	0.567	0.7462	0.8059	0.6395	0.7448	0.7963	0.5984	0.7467
phoneme-1red_vs_0red	0.8529	0.7399	0.7984	0.8556	0.7235	0.7984	0.8540	0.7058	0.7996
segment-5red_vs_1-2-3	0.9088	0.765	0.9259	0.8907	0.8264	0.9299	0.9062	0.8227	0.9269
segment-6red_vs_3-4-5	0.9972	0.9899	0.9997	0.9976	0.9874	0.9997	0.9962	0.9904	0.9997
segment-7red_vs_2-4-5-6	1,0000	0.9996	1,0000	1,0000	0.9992	1,0000	1,0000	0.9996	1,0000
segment0	0.9994	0.9896	0.9986	0.9993	0.9965	0.9986	0.9995	0.9906	0.9985
shuttle-2_vs_1red	0.9978	0.9993	0.9809	0.9985	0.9994	0.9811	0.9984	0.9996	0.9807
shuttle-2_vs_5	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0.9991	1,0000
shuttle-6-7_vs_1redB	1,0000	0.9844	0.9516	1,0000	0.9844	0.9528	0.9999	0.9800	0.9464
shuttle-6_vs_2-3	0.9432	0.9977	1,0000	0.9932	0.9977	1,0000	1,0000	0.9977	1,0000
shuttle-c0-vs-c4	0.9960	0.9997	0.992	0.9960	1,0000	0.9920	0.9960	0.9997	0.9920
shuttle-c2-vs-c4	1,0000	0.9958	1,0000	1,0000	0.9958	1,0000	1,0000	1,0000	0.9960
texture-12red_vs_13-14	1,0000	0.9975	1,0000	1,0000	0.9980	1,0000	1,0000	0.9995	1,0000
texture-2red_vs_3-4	0.9934	0.8964	0.9838	0.9921	0.8603	0.9842	0.9932	0.8878	0.9821
texture-6red_vs_7-8	1,0000	0.9273	1,0000	1,0000	0.9215	1,0000	1,0000	0.9245	1,0000
texture-7red_vs_2-3-4-6	1,0000	0.925	1,0000	1,0000	0.9250	1,0000	1,0000	0.9250	1,0000
vehicle0	0.9662	0.9576	0.9892	0.9653	0.9472	0.9882	0.9649	0.9423	0.9857
vehicle1	0.7868	0.707	0.8306	0.7782	0.6904	0.8305	0.7784	0.7695	0.8176
vehicle2	0.9710	0.9505	0.9727	0.9683	0.9532	0.9735	0.9679	0.9697	0.9735
vehicle3	0.7821	0.7195	0.7822	0.7752	0.6935	0.7851	0.7677	0.7029	0.7625
wdbc-MredB_vs_B	0.9749	0.7899	0.9925	0.9796	0.8243	0.9962	0.9847	0.8577	0.9981
wdbc-Mred_vs_B	0.9951	0.8345	0.9986	1,0000	0.8930	1,0000	0.9965	0.8930	0.9972
winequality-red-3_vs_5	0.8199	0.6449	0.8077	0.8155	0.4927	0.8077	0.8067	0.6041	0.8210
winequality-red-4	0.7164	0.5545	0.737	0.7241	0.6181	0.7326	0.7205	0.5709	0.7299
winequality-red-8_vs_6	0.7596	0.696	0.8842	0.7651	0.6707	0.8821	0.7344	0.6946	0.8779
winequality-red-8_vs_6-7	0.6788	0.6399	0.8314	0.6889	0.5332	0.8268	0.6587	0.6498	0.8257
winequality-white-3-9_vs_5	0.7276	0.6105	0.697	0.7288	0.5400	0.7002	0.7217	0.5569	0.6830
winequality-white-3_vs_7	0.8884	0.6121	0.7341	0.8732	0.6007	0.7315	0.8717	0.6311	0.7241
winequality-white-9_vs_4	0.9908	0.6237	0.9333	0.9939	0.5964	0.9394	0.9362	0.7082	0.9273
wisconsinImb	0.9924	0.967	0.995	0.9927	0.9694	0.9950	0.9915	0.9514	0.9954
yeast-0-2-5-6_vs_3-7-8-9	0.7983	0.7494	0.8391	0.8008	0.7822	0.8386	0.7988	0.7624	0.8410
yeast-0-2-5-7-9_vs_3-6-8	0.9532	0.9107	0.9394	0.9562	0.9028	0.9390	0.9557	0.9248	0.9392
yeast-0-3-5-9_vs_7-8	0.8116	0.7603	0.8092	0.8142	0.7261	0.8092	0.8184	0.7357	0.8098
yeast-0-5-6-7-9_vs_4	0.8926	0.8346	0.8552	0.8968	0.7907	0.8547	0.8962	0.7471	0.8554
yeast-1-2-8-9_vs_7	0.7422	0.6774	0.815	0.7519	0.5396	0.8143	0.7478	0.6400	0.8127
yeast-1-4-5-8_vs_7	0.7000	0.6163	0.7066	0.6902	0.5257	0.7055	0.6880	0.5889	0.7086
yeast-1_vs_7	0.8121	0.7164	0.8499	0.8202	0.6664	0.8487	0.8033	0.7174	0.8480
yeast-2_vs_4	0.9650	0.8789	0.9415	0.9723	0.9105	0.9413	0.9659	0.9097	0.9411
yeast-2_vs_8	0.8678	0.8263	0.853	0.8718	0.8010	0.7590	0.8862	0.8703	0.8486
yeast1	0.7920	0.7291	0.787	0.7921	0.7400	0.7877	0.7928	0.7095	0.7880
yeast3	0.9725	0.9104	0.9685	0.9722	0.8845	0.9681	0.9724	0.8935	0.9677

yeast4	0.9206	0.7478	0.8723	0.9158	0.7178	0.8719	0.9178	0.7709	0.8723
yeast5	0.9909	0.9529	0.9886	0.9903	0.9376	0.9876	0.9915	0.9837	0.9876
yeast6	0.9241	0.8295	0.9347	0.9209	0.8225	0.9357	0.9228	0.8122	0.9361
Mean	0.9096	0.8315	0.9000	0.9085	0.8266	0.9001	0.9084	0.8412	0.9005

La tabla E.2 muestra los resultados de AUC en cuanto a test para los algoritmos basados en aprendizaje sensible al coste (CS-C4.5 y CS-SVM), así como el algoritmo ensemble EUSBOOST.

Tabla E.2: Resultados de AUC para los algoritmos cost sensitive y el ensemble EUSBOOST

Datasets	**CS-C4.5**	**CS-SVM**	**EUSBOOST**
abalone-17_vs_7-8-9-10	0.6932	0.9459	0.9003
abalone-19_vs_10-11-12-13	0.5339	0.7128	0.5933
abalone-20_vs_8-9-10	0.7142	0.9773	0.8994
abalone-21_vs_8	0.8564	0.9438	0.8547
abalone-3_vs_11	0.9656	1.0000	0.9656
abalone19	0.5692	0.8577	0.7779
abalone9-18	0.6703	0.9616	0.8626
appendicitisImb	0.7059	0.8294	0.8526
cleveland-0_vs_4	0.6797	0.9480	0.9239
cleveland-4	0.6712	0.8415	0.7598
ecoli-0-1-3-7_vs_2-6	0.8336	0.9199	0.8263
ecoli-0-1-4-6_vs_5	0.8490	0.8740	0.9615
ecoli-0-1-4-7_vs_2-3-5-6	0.8780	0.9107	0.9582
ecoli-0-1-4-7_vs_5-6	0.8509	0.9130	0.9480
ecoli-0-1_vs_2-3-5	0.7661	0.9361	0.9539
ecoli-0-1_vs_5	0.8182	0.8966	0.9852
ecoli-0-2-3-4_vs_5	0.8334	0.8577	0.9825
ecoli-0-2-6-7_vs_3-5	0.8496	0.9051	0.8852
ecoli-0-3-4-6_vs_5	0.8588	0.8365	0.9439
ecoli-0-3-4-7_vs_5-6	0.7837	0.9146	0.9729
ecoli-0-3-4_vs_5	0.9257	0.8917	0.9854
ecoli-0-4-6_vs_5	0.8365	0.8689	0.9919
ecoli-0-6-7_vs_3-5	0.8850	0.8438	0.9046
ecoli-0-6-7_vs_5	0.8913	0.8800	0.9469
ecoli-0_vs_1	0.9832	0.9926	0.9832
ecoli1	0.9163	0.9504	0.9646
ecoli2	0.8883	0.9341	0.9424
ecoli3	0.8315	0.9406	0.9235
ecoli4	0.8613	0.9952	0.9754
glass-0-1-2-3_vs_4-5-6	0.8536	0.9533	0.9744
glass-0-1-4-6_vs_2	0.6815	0.6423	0.8117
glass-0-1-5_vs_2	0.5992	0.5210	0.6898
glass-0-1-6_vs_2	0.6114	0.6014	0.7762
glass-0-1-6_vs_5	0.9886	0.9943	0.9943
glass-0-4_vs_5	0.9941	0.9875	0.9941
glass-0-6_vs_5	0.9950	0.9947	0.9950
glass0	0.8365	0.8399	0.9089
glass1	0.7529	0.6681	0.8682
glass2	0.6412	0.6556	0.7376

glass4	0.8431	0.8896	0.9290
glass5	0.9427	0.9951	0.9939
glass6	0.9018	0.9045	0.9480
habermanImb	0.6575	0.6988	0.6451
ionosphere-bredB_vs_g	0.4667	0.9733	0.8611
ionosphere-bred_vs_g	0.6400	0.7267	0.8944
iris0	0.9900	1.0000	0.9900
kddcup-buffer_overflow_vs_back	0.9833	1.0000	0.9994
kddcup-guess_passwd_vs_satan	0.9991	0.9994	0.9997
kddcup-land_vs_portsweep	1.0000	0.9990	1.0000
kddcup-land_vs_satan	1.0000	1.0000	1.0000
kddcup-rootkit-imap_vs_back	0.9800	1.0000	0.9794
magic-hredB_vs_gredB	0.7857	0.7118	0.8421
magic-hred_vs_gred	0.7648	0.8255	0.8823
movement_libras-1	0.8531	0.9550	0.9222
new-thyroid1	0.9746	0.9992	0.9980
new-thyroid2	0.9802	1.0000	0.9952
page-blocks-1-3_vs_4	0.9789	0.9897	0.9987
page-blocks0	0.9522	0.9584	0.9874
phoneme-1redB_vs_0redB	0.5964	0.7508	0.7662
phoneme-1red_vs_0red	0.6475	0.8002	0.8730
segment-5red_vs_1-2-3	0.8699	0.9356	0.9321
segment-6red_vs_3-4-5	1.0000	1.0000	0.9904
segment-7red_vs_2-4-5-6	1.0000	1.0000	1.0000
segment0	0.9917	0.9999	0.9993
shuttle-2_vs_1red	0.9998	0.9993	0.9995
shuttle-2_vs_5	1.0000	0.9999	1.0000
shuttle-6-7_vs_1redB	0.9798	0.9178	1.0000
shuttle-6_vs_2-3	0.9977	1.0000	0.9977
shuttle-c0-vs-c4	0.9997	1.0000	1.0000
shuttle-c2-vs-c4	1.0000	1.0000	1.0000
texture-12red_vs_13-14	0.9333	1.0000	1.0000
texture-2red_vs_3-4	0.8844	0.9937	0.9627
texture-6red_vs_7-8	0.9430	1.0000	0.9991
texture-7red_vs_2-3-4-6	1.0000	1.0000	0.9947
vehicle0	0.9351	0.9919	0.9917
vehicle1	0.6814	0.8776	0.8278
vehicle2	0.9501	0.9905	0.9965
vehicle3	0.7276	0.8511	0.8501
wdbc-MredB_vs_B	0.8377	0.9777	0.9786
wdbc-Mred_vs_B	0.8847	0.9394	0.9697
winequality-red-3_vs_5	0.7151	0.7995	0.8335
winequality-red-4	0.5860	0.7195	0.6746
winequality-red-8_vs_6	0.6354	0.8379	0.8768
winequality-red-8_vs_6-7	0.6298	0.8024	0.7503
winequality-white-3-9_vs_5	0.6378	0.6788	0.7203
winequality-white-3_vs_7	0.5896	0.7338	0.7807
winequality-white-9_vs_4	0.7814	0.8424	0.8330
wisconsinImb	0.9695	0.9948	0.9875
yeast-0-2-5-6_vs_3-7-8-9	0.7922	0.8411	0.8631
yeast-0-2-5-7-9_vs_3-6-8	0.9038	0.9388	0.9705
yeast-0-3-5-9_vs_7-8	0.6900	0.8065	0.7932
yeast-0-5-6-7-9_vs_4	0.7255	0.8528	0.8742

yeast-1-2-8-9_vs_7	0.6801	0.8121	0.7843
yeast-1-4-5-8_vs_7	0.5556	0.6998	0.5853
yeast-1_vs_7	0.6154	0.8503	0.8501
yeast-2_vs_4	0.8879	0.9383	0.9701
yeast-2_vs_8	0.8685	0.8493	0.8563
yeast1	0.7241	0.7571	0.7770
yeast3	0.9171	0.9681	0.9682
yeast4	0.7270	0.8754	0.9325
yeast5	0.9334	0.9871	0.9856
yeast6	0.8135	0.9370	0.9258
Mean	0.8263	0.8952	0.9094

ANEXO F

G-Mean obtenido por los algoritmos IFROWANN

En este apéndice se presentan los resultados de G-Mean obtenidos por los algoritmos basados en IFROWANN introducidos en el Capítulo 4. Se muestran los resultados usando las 6 estrategias de construcción de vectores de ponderación para cada una de las 3 relaciones de separabilidad utilizadas en esta tesis: Lukasiewicz (Tabla F.1), Average (Tabla F.2) y Minimum (Tabla F.3).

Tabla F.1: Resultados de G-Mean para los algoritmos IFROWANN usando Lukasiewicz

Datasets	**VCCAD**	$\mathscr{W}_1$	$\mathscr{W}_2$	$\mathscr{W}_3$	$\mathscr{W}_4$	$\mathscr{W}_5$	$\mathscr{W}_6$
abalone-17_vs_7-8-9-10	0.4912	0.7040	0.4008	0.0000	0.4251	0.1758	0.8534
abalone-19_vs_10-11-12-13	0.0745	0.6247	0.0708	0.0000	0.0000	0.0000	0.6220
abalone-20_vs_8-9-10	0.3339	0.7427	0.4909	0.0000	0.2530	0.0000	0.8504
abalone-21_vs_8	0.6504	0.8162	0.6697	0.0000	0.5357	0.5828	0.7899
abalone-3_vs_11	1,0000	0.9896	0.9886	1,0000	1,0000	0.9979	0.9959
abalone19	0.0755	0.6384	0.4122	0.0000	0.0000	0.0000	0.6033
abalone9-18	0.6374	0.7035	0.0381	0.0000	0.5073	0.0707	0.7916
appendicitisImb	0.6436	0.7440	0.6087	0.6678	0.7044	0.7492	0.6940
cleveland-0_vs_4	0.7480	0.7480	0.7480	0.7480	0.7480	0.7480	0.7480
cleveland-4	0.5108	0.5013	0.4996	0.5142	0.5142	0.5098	0.5060
ecoli-0-1-3-7_vs_2-6	0.7309	0.6910	0.6684	0.7365	0.7365	0.7365	0.7365
ecoli-0-1-4-6_vs_5	0.8823	0.9557	0.9402	0.7799	0.9088	0.9088	0.9577
ecoli-0-1-4-7_vs_2-3-5-6	0.8401	0.9045	0.7668	0.4529	0.8470	0.8297	0.8656
ecoli-0-1-4-7_vs_5-6	0.9056	0.8963	0.8636	0.4149	0.8847	0.8862	0.9002
ecoli-0-1_vs_2-3-5	0.7994	0.8595	0.7649	0.7093	0.8089	0.8534	0.9214
ecoli-0-1_vs_5	0.8689	0.9289	0.8855	0.8176	0.8689	0.9126	0.9548

ecoli-0-2-3-4_vs_5	0.8759	0.9435	0.9434	0.8205	0.9023	0.8995	0.9487
ecoli-0-2-6-7_vs_3-5	0.8698	0.8830	0.8632	0.3732	0.8720	0.8985	0.9090
ecoli-0-3-4-6_vs_5	0.9094	0.9520	0.9554	0.7640	0.9094	0.9118	0.9572
ecoli-0-3-4-7_vs_5-6	0.9008	0.8955	0.8308	0.5830	0.8841	0.8841	0.9128
ecoli-0-3-4_vs_5	0.8758	0.9464	0.9486	0.8147	0.9067	0.9091	0.9515
ecoli-0-4-6_vs_5	0.8772	0.9519	0.9553	0.7831	0.8800	0.9095	0.9543
ecoli-0-6-7_vs_3-5	0.8811	0.9029	0.8240	0.3159	0.8546	0.9098	0.9229
ecoli-0-6-7_vs_5	0.8542	0.8743	0.8366	0.4146	0.8271	0.8560	0.8837
ecoli-0_vs_1	0.9652	0.9721	0.9149	0.8851	0.9794	0.9590	0.9443
ecoli1	0.7848	0.8985	0.8038	0.2841	0.8427	0.8549	0.8599
ecoli2	0.8811	0.8929	0.5519	0.6239	0.9414	0.9242	0.8759
ecoli3	0.7509	0.8479	0.8105	0.0000	0.7380	0.8592	0.8717
ecoli4	0.8551	0.8970	0.8610	0.6132	0.8150	0.8896	0.9268
glass-0-1-2-3_vs_4-5-6	0.9372	0.9133	0.9291	0.8054	0.9372	0.8896	0.9449
glass-0-1-4-6_vs_2	0.3938	0.5276	0.4701	0.0000	0.1931	0.0000	0.6304
glass-0-1-5_vs_2	0.5173	0.6474	0.3166	0.0000	0.0000	0.0000	0.6407
glass-0-1-6_vs_2	0.4885	0.4766	0.4129	0.0000	0.2496	0.0000	0.7007
glass-0-1-6_vs_5	0.8431	0.9471	0.9410	0.6548	0.9030	0.9149	0.9119
glass-0-4_vs_5	0.9817	0.7681	0.7552	0.8682	0.9249	0.9817	0.9817
glass-0-6_vs_5	0.9585	0.9585	0.9475	0.9639	0.9585	0.9585	0.9585
glass0	0.7900	0.6121	0.6166	0.0000	0.8171	0.7886	0.5696
glass1	0.7528	0.6666	0.4637	0.1434	0.7690	0.5544	0.5100
glass2	0.5217	0.5355	0.4765	0.0000	0.0000	0.0000	0.5993
glass4	0.8971	0.9540	0.9408	0.6442	0.8971	0.8924	0.8874
glass5	0.7749	0.8467	0.8928	0.7237	0.7749	0.7723	0.8966
glass6	0.9160	0.9325	0.9215	0.9187	0.9160	0.9350	0.9325
habermanImb	0.4669	0.5641	0.0000	0.0000	0.4167	0.0500	0.0422
ionosphere-bredB_vs_g	0.5942	0.0000	0.0000	0.5942	0.5942	0.5942	0.5942
ionosphere-bred_vs_g	0.6059	0.0000	0.0000	0.6059	0.6059	0.6059	0.6059
iris0	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-buffer_overflow_vs_back	0.9450	0.9794	0.9507	0.6279	0.9080	0.9462	0.9902
kddcup-guess_passwd_vs_satan	0.9984	0.9984	0.9894	0.9984	0.9984	0.9984	0.9984
kddcup-land_vs_portsweep	0.9971	0.8041	0.8041	0.9971	0.9971	0.8870	0.8870
kddcup-land_vs_satan	0.9984	0.8882	0.8882	0.9984	0.9984	0.9984	0.9984
kddcup-rootkit-imap_vs_back	0.9192	0.9721	0.9666	0.8981	0.9190	0.8725	0.8992
magic-hredB_vs_gredB	0.4225	0.7157	0.5548	0.4240	0.4240	0.2976	0.4703
magic-hred_vs_gred	0.4442	0.7368	0.4459	0.3624	0.4453	0.4491	0.6687
movement_libras-1	0.3914	0.1592	0.1592	0.3914	0.3914	0.0756	0.0756
new-thyroid1	0.9768	0.9084	0.9423	0.6849	0.9647	0.8023	0.9831
new-thyroid2	0.9796	0.9067	0.9240	0.6811	0.9647	0.7974	0.9859
page-blocks-1-3_vs_4	0.9966	0.9416	0.7686	0.6857	0.9792	0.5991	0.9703
page-blocks0	0.8793	0.7831	0.2167	0.2362	0.8815	0.6637	0.7839
phoneme-1redB_vs_0redB	0.3785	0.7466	0.4305	0.0000	0.0000	0.0000	0.7197
phoneme-1red_vs_0red	0.3745	0.7860	0.4764	0.0000	0.1839	0.0000	0.7880
segment-5red_vs_1-2-3	0.7857	0.9031	0.7859	0.4291	0.6643	0.6940	0.8959
segment-6red_vs_3-4-5	0.5935	0.9587	0.9396	0.0000	0.4790	0.6973	0.9259
segment-7red_vs_2-4-5-6	0.9981	1,0000	1,0000	0.9981	0.9981	0.9981	0.9981
segment0	0.9929	0.9723	0.8921	0.7765	0.9914	0.9901	0.9350
shuttle-2_vs_1red	0.9884	0.8397	0.6516	0.6487	0.9577	0.7188	0.9971
shuttle-2_vs_5	1,0000	0.9140	0.8725	0.8302	1,0000	0.8424	0.9997
shuttle-6-7_vs_1redB	0.9732	0.7541	0.9520	0.3683	0.8859	0.5054	1,0000
shuttle-6_vs_2-3	0.8812	0.9479	0.8912	0.2828	0.8828	0.6828	0.8828
shuttle-c0-vs-c4	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

shuttle-c2-vs-c4	0.9374	1,0000	1,0000	0.9374	0.9374	0.9374	0.9374
texture-12red_vs_13-14	0.8677	0.9633	0.9633	0.8677	0.8677	0.8677	0.8677
texture-2red_vs_3-4	0.9458	0.6635	0.6615	0.9458	0.9458	0.6654	0.6635
texture-6red_vs_7-8	0.9664	0.7679	0.7679	0.9664	0.9664	0.8348	0.8348
texture-7red_vs_2-3-4-6	0.9912	0.8695	0.8695	0.9912	0.9912	0.9912	0.9912
vehicle0	0.9186	0.9325	0.8953	0.7148	0.9267	0.9253	0.9021
vehicle1	0.5523	0.6793	0.6147	0.3348	0.5826	0.6072	0.6523
vehicle2	0.9488	0.9386	0.8963	0.7673	0.9512	0.9533	0.9050
vehicle3	0.6161	0.6839	0.6550	0.3158	0.6080	0.6245	0.6723
wdbc-MredB_vs_B	0.6583	0.6583	0.6583	0.6583	0.6583	0.6583	0.6583
wdbc-Mred_vs_B	0.6540	0.6540	0.6540	0.6540	0.6540	0.6540	0.6540
winequality-red-3_vs_5	0.0000	0.4553	0.5295	0.0000	0.0000	0.2797	0.4180
winequality-red-4	0.3151	0.6448	0.4137	0.1233	0.1830	0.1228	0.6610
winequality-red-8_vs_6	0.4917	0.7539	0.6819	0.0000	0.0992	0.0972	0.7080
winequality-red-8_vs_6-7	0.1982	0.6594	0.6425	0.0000	0.0000	0.0000	0.5361
winequality-white-3-9_vs_5	0.2156	0.5038	0.3743	0.0894	0.1265	0.2680	0.4784
winequality-white-3_vs_7	0.3407	0.3300	0.5122	0.3414	0.3414	0.0000	0.0000
winequality-white-9_vs_4	0.7813	0.7217	0.6907	0.5936	0.7843	0.7843	0.7813
wisconsinImb	0.9748	0.7590	0.9710	0.7457	0.9728	0.7538	0.8124
yeast-0-2-5-6_vs_3-7-8-9	0.7448	0.8047	0.0914	0.1800	0.7369	0.6364	0.5903
yeast-0-2-5-7-9_vs_3-6-8	0.8813	0.9183	0.2094	0.2501	0.8875	0.8826	0.8431
yeast-0-3-5-9_vs_7-8	0.6007	0.7381	0.2100	0.1527	0.5285	0.4395	0.7297
yeast-0-5-6-7-9_vs_4	0.6112	0.8218	0.4596	0.0000	0.5928	0.4730	0.7868
yeast-1-2-8-9_vs_7	0.3244	0.7069	0.1994	0.0000	0.2447	0.0000	0.6634
yeast-1-4-5-8_vs_7	0.3082	0.6168	0.1856	0.0000	0.0816	0.0000	0.6844
yeast-1_vs_7	0.5534	0.7594	0.2772	0.0000	0.4378	0.0816	0.7852
yeast-2_vs_4	0.8296	0.8621	0.6040	0.2884	0.8064	0.7171	0.9338
yeast-2_vs_8	0.7236	0.7601	0.3976	0.6657	0.6965	0.6975	0.7263
yeast1	0.6247	0.6877	0.1071	0.0000	0.6207	0.3687	0.1239
yeast3	0.7959	0.8919	0.1491	0.0000	0.8118	0.7606	0.7660
yeast4	0.5824	0.8444	0.4163	0.0000	0.4912	0.4099	0.8460
yeast5	0.7932	0.9505	0.9079	0.2040	0.8079	0.9397	0.9589
yeast6	0.6849	0.8745	0.5543	0.0000	0.7063	0.8406	0.8940
Mean	0.7280	0.7858	0.6541	0.4622	0.6864	0.6309	0.7886

Tabla F.2: Resultados de la G-Mean para los algoritmos IFROWANN usando Average

Datasets	**VCCAD**	$\mathscr{W}_1$	$\mathscr{W}_2$	$\mathscr{W}_3$	$\mathscr{W}_4$	$\mathscr{W}_5$	$\mathscr{W}_6$
abalone-17_vs_7-8-9-10	0.4912	0.6884	0.3379	0.0000	0.4251	0.0000	0.8534
abalone-19_vs_10-11-12-13	0.0745	0.6173	0.0000	0.0000	0.0000	0.0000	0.6220
abalone-20_vs_8-9-10	0.3339	0.7439	0.4718	0.0000	0.2530	0.0000	0.8504
abalone-21_vs_8	0.6504	0.8096	0.7230	0.0000	0.5357	0.5835	0.7899
abalone-3_vs_11	1,0000	0.9802	0.9759	1,0000	1,0000	0.9979	0.9959
abalone19	0.0755	0.6423	0.2985	0.0000	0.0000	0.0000	0.6025
abalone9-18	0.6374	0.6857	0.0000	0.0000	0.5073	0.0000	0.7916
appendicitisImb	0.6541	0.7917	0.5790	0.4279	0.7241	0.7224	0.7270
cleveland-0_vs_4	0.6230	0.9530	0.8971	0.0000	0.5724	0.6309	0.8249
cleveland-4	0.1389	0.7504	0.7001	0.0000	0.0000	0.0000	0.5948
ecoli-0-1-3-7_vs_2-6	0.7346	0.7290	0.6908	0.7401	0.7401	0.7383	0.7383
ecoli-0-1-4-6_vs_5	0.8541	0.9646	0.8742	0.7146	0.8809	0.8809	0.9259
ecoli-0-1-4-7_vs_2-3-5-6	0.7995	0.8666	0.7166	0.1633	0.8060	0.7384	0.9193
ecoli-0-1-4-7_vs_5-6	0.8845	0.8924	0.7958	0.1789	0.8636	0.7331	0.9019

ecoli-0-1_vs_2-3-5	0.7713	0.8998	0.8461	0.6258	0.7585	0.7887	0.9051
ecoli-0-1_vs_5	0.8421	0.9507	0.8969	0.7560	0.8441	0.8878	0.9355
ecoli-0-2-3-4_vs_5	0.8541	0.9427	0.8821	0.7560	0.8854	0.8835	0.9284
ecoli-0-2-6-7_vs_3-5	0.8381	0.8139	0.8252	0.0000	0.7767	0.7296	0.8474
ecoli-0-3-4-6_vs_5	0.8855	0.9526	0.8559	0.7293	0.8855	0.8851	0.9303
ecoli-0-3-4-7_vs_5-6	0.8747	0.8873	0.7737	0.1789	0.8579	0.7921	0.9173
ecoli-0-3-4_vs_5	0.8521	0.9464	0.8816	0.7293	0.8859	0.8854	0.9335
ecoli-0-4-6_vs_5	0.8410	0.9374	0.8764	0.7146	0.8437	0.8718	0.9283
ecoli-0-6-7_vs_3-5	0.8530	0.8449	0.8236	0.0000	0.6997	0.5679	0.8589
ecoli-0-6-7_vs_5	0.8277	0.8400	0.8384	0.1000	0.8003	0.8293	0.8585
ecoli-0_vs_1	0.9652	0.9590	0.9149	0.8697	0.9794	0.9390	0.9443
ecoli1	0.7863	0.8916	0.7202	0.2684	0.8444	0.8498	0.8612
ecoli2	0.8828	0.8722	0.3479	0.6253	0.9431	0.9208	0.8642
ecoli3	0.7524	0.8513	0.7923	0.0000	0.7393	0.8641	0.8717
ecoli4	0.8567	0.9052	0.8603	0.6146	0.8166	0.8912	0.9269
glass-0-1-2-3_vs_4-5-6	0.9329	0.8291	0.9121	0.6465	0.9257	0.7888	0.9451
glass-0-1-4-6_vs_2	0.4094	0.4833	0.4096	0.0000	0.1986	0.0000	0.6550
glass-0-1-5_vs_2	0.5263	0.5994	0.0359	0.0000	0.0000	0.0000	0.6907
glass-0-1-6_vs_2	0.5024	0.4810	0.3508	0.0000	0.2589	0.0000	0.7255
glass-0-1-6_vs_5	0.7307	0.8838	0.8610	0.5414	0.7942	0.8759	0.9797
glass-0-4_vs_5	0.9354	0.9350	0.9155	0.6243	0.9372	0.9414	0.9813
glass-0-6_vs_5	0.9328	0.8876	0.8282	0.6828	0.9949	0.9949	0.9899
glass0	0.8188	0.5826	0.4195	0.0000	0.8504	0.8255	0.5508
glass1	0.7715	0.5967	0.3764	0.0000	0.7870	0.3009	0.4505
glass2	0.5442	0.5055	0.4054	0.0000	0.0000	0.0000	0.6227
glass4	0.8551	0.8465	0.8298	0.2788	0.8551	0.8068	0.8707
glass5	0.7315	0.8719	0.8436	0.1414	0.7365	0.7379	0.9139
glass6	0.9232	0.8804	0.9244	0.8725	0.8893	0.8650	0.9055
habermanImb	0.4669	0.5711	0.0000	0.0000	0.4167	0.0000	0.0422
ionosphere-bredB_vs_g	0.3398	0.9221	0.9159	0.0000	0.0000	0.3398	0.3382
ionosphere-bred_vs_g	0.4227	0.6182	0.8185	0.1414	0.4243	0.2828	0.4243
iris0	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-buffer_overflow_vs_back	0.9240	0.9511	0.9109	0.3939	0.8913	0.9097	0.9890
kddcup-guess_passwd_vs_satan	0.9907	0.9860	0.9654	0.9804	0.9907	0.9804	0.9907
kddcup-land_vs_portsweep	1,0000	0.9913	0.9898	0.9789	0.9789	0.9995	0.9961
kddcup-land_vs_satan	1,0000	0.9892	0.9829	0.9789	0.9789	0.9789	1,0000
kddcup-rootkit-imap_vs_back	0.9196	0.8515	0.9670	0.7042	0.8985	0.7253	0.9456
magic-hredB_vs_gredB	0.3855	0.5927	0.4706	0.0000	0.3869	0.2682	0.6431
magic-hred_vs_gred	0.4168	0.6357	0.3428	0.0000	0.3766	0.2228	0.6987
movement_libras-1	0.8859	0.9488	0.5595	0.0000	0.8408	0.5689	0.9505
new-thyroid1	0.9768	0.8923	0.9423	0.6849	0.9647	0.7821	0.9831
new-thyroid2	0.9796	0.9067	0.9240	0.6811	0.9647	0.7635	0.9859
page-blocks-1-3_vs_4	0.9789	0.8156	0.6618	0.1633	0.9615	0.6547	0.9807
page-blocks0	0.8793	0.6673	0.2058	0.0000	0.8805	0.4521	0.7835
phoneme-1redB_vs_0redB	0.3785	0.7400	0.4097	0.0000	0.0000	0.0000	0.7195
phoneme-1red_vs_0red	0.3745	0.7809	0.4594	0.0000	0.1839	0.0000	0.7880
segment-5red_vs_1-2-3	0.7890	0.5877	0.5782	0.0000	0.4891	0.3789	0.8550
segment-6red_vs_3-4-5	0.5977	0.8201	0.7881	0.0000	0.3044	0.6566	0.9934
segment-7red_vs_2-4-5-6	1,0000	0.9317	0.9078	1,0000	1,0000	1,0000	0.9996
segment0	0.9975	0.6778	0.4194	0.1471	0.9959	0.9799	0.8530
shuttle-2_vs_1red	0.9886	0.8397	0.6516	0.6488	0.9578	0.7189	0.9972
shuttle-2_vs_5	1,0000	0.9140	0.8721	0.8302	1,0000	0.8424	0.9997
shuttle-6-7_vs_1redB	0.9732	0.7541	0.9520	0.0894	0.8859	0.2789	1,0000

shuttle-6_vs_2-3	0.8812	0.9456	0.8912	0.2828	0.8828	0.6828	0.8828
shuttle-c0-vs-c4	0.9960	1,0000	1,0000	0.9960	0.9960	0.9960	1,0000
shuttle-c2-vs-c4	0.9414	1,0000	1,0000	0.9414	0.9414	0.9414	0.9414
texture-12red_vs_13-14	1,0000	0.9864	0.9772	0.9633	1,0000	0.9995	0.9995
texture-2red_vs_3-4	0.8726	0.7647	0.0761	0.0000	0.8575	0.6302	0.9636
texture-6red_vs_7-8	1,0000	0.8233	0.7217	0.9146	1,0000	0.9732	1,0000
texture-7red_vs_2-3-4-6	1,0000	0.8662	0.8427	1,0000	1,0000	1,0000	1,0000
vehicle0	0.9374	0.7121	0.5611	0.0000	0.9558	0.6585	0.6980
vehicle1	0.5388	0.6388	0.0000	0.0000	0.5752	0.1517	0.0615
vehicle2	0.9689	0.6623	0.0000	0.0000	0.9751	0.6111	0.4344
vehicle3	0.6159	0.6490	0.0000	0.0000	0.5755	0.0617	0.2551
wdbc-MredB_vs_B	0.7942	0.8873	0.9279	0.2309	0.6421	0.5942	0.8899
wdbc-Mred_vs_B	0.8828	0.8828	0.9330	0.6828	0.6828	0.8828	0.8828
winequality-red-3_vs_5	0.0000	0.7424	0.7036	0.0000	0.0000	0.0000	0.2808
winequality-red-4	0.2284	0.6604	0.0114	0.0000	0.0602	0.0000	0.6652
winequality-red-8_vs_6	0.4946	0.7852	0.6286	0.0000	0.0996	0.0000	0.6928
winequality-red-8_vs_6-7	0.1988	0.6911	0.5389	0.0000	0.0000	0.0000	0.5817
winequality-white-3-9_vs_5	0.1786	0.4509	0.3511	0.0000	0.0894	0.0000	0.4512
winequality-white-3_vs_7	0.1000	0.2755	0.6085	0.0000	0.1000	0.0000	0.1989
winequality-white-9_vs_4	0.7939	0.7186	0.7009	0.4000	0.8000	0.7969	0.7969
wisconsinImb	0.9636	0.9277	0.9728	0.5508	0.9646	0.9254	0.9775
yeast-0-2-5-6_vs_3-7-8-9	0.7448	0.7740	0.0000	0.0000	0.7369	0.4557	0.5893
yeast-0-2-5-7-9_vs_3-6-8	0.8813	0.8692	0.0900	0.0447	0.8875	0.8290	0.8399
yeast-0-3-5-9_vs_7-8	0.6007	0.7380	0.0000	0.0000	0.5285	0.4149	0.7244
yeast-0-5-6-7-9_vs_4	0.6112	0.8152	0.4418	0.0000	0.5936	0.4729	0.7819
yeast-1-2-8-9_vs_7	0.3244	0.7002	0.0612	0.0000	0.2447	0.0000	0.6612
yeast-1-4-5-8_vs_7	0.3082	0.6149	0.0593	0.0000	0.0816	0.0000	0.6828
yeast-1_vs_7	0.5561	0.7634	0.1881	0.0000	0.3593	0.0816	0.7865
yeast-2_vs_4	0.8296	0.8651	0.6031	0.2421	0.8072	0.7053	0.9338
yeast-2_vs_8	0.7236	0.7283	0.4079	0.6657	0.6965	0.6657	0.7268
yeast1	0.6247	0.6758	0.0000	0.0000	0.6207	0.3630	0.0138
yeast3	0.7959	0.8818	0.0174	0.0000	0.8118	0.7606	0.7644
yeast4	0.5824	0.8324	0.3865	0.0000	0.4912	0.3647	0.8438
yeast5	0.7932	0.9483	0.9029	0.1374	0.8079	0.9394	0.9571
yeast6	0.6849	0.8701	0.5355	0.0000	0.7063	0.8401	0.8913
Mean	0.7192	0.7993	0.6093	0.3084	0.6612	0.5719	0.7916

Tabla F.3: Resultados de la G-Mean para los algoritmos IFROWANN usando Minimun

Datasets	**VCCAD**	$\mathscr{W}_1$	$\mathscr{W}_2$	$\mathscr{W}_3$	$\mathscr{W}_4$	$\mathscr{W}_5$	$\mathscr{W}_6$
abalone-17_vs_7-8-9-10	0.5265	0.7705	0.3745	0.0000	0.3245	0.1180	0.8607
abalone-19_vs_10-11-12-13	0.0809	0.6576	0.0000	0.0000	0.0000	0.0000	0.6593
abalone-20_vs_8-9-10	0.3625	0.8163	0.4724	0.0000	0.3346	0.0894	0.8341
abalone-21_vs_8	0.7456	0.8298	0.7719	0.3724	0.6506	0.6500	0.7904
abalone-3_vs_11	1,0000	0.9907	0.9886	1,0000	1,0000	1,0000	0.9979
abalone19	0.0000	0.6748	0.3370	0.0000	0.0000	0.0000	0.6618
abalone9-18	0.7210	0.7689	0.0000	0.0000	0.5065	0.2593	0.8326
appendicitisImb	0.7422	0.7360	0.4995	0.1970	0.7260	0.7006	0.7443
cleveland-0_vs_4	0.7920	0.8542	0.8233	0.4138	0.6013	0.5453	0.8637
cleveland-4	0.3624	0.7780	0.7494	0.1155	0.1155	0.3642	0.6175
ecoli-0-1-3-7_vs_2-6	0.7309	0.7365	0.7140	0.7365	0.7365	0.7365	0.7365
ecoli-0-1-4-6_vs_5	0.8524	0.8568	0.7321	0.6560	0.8541	0.7862	0.8906

ecoli-0-1-4-7_vs_2-3-5-6	0.8402	0.9175	0.6260	0.1971	0.8662	0.5128	0.8872
ecoli-0-1-4-7_vs_5-6	0.8609	0.8844	0.7191	0.0894	0.8354	0.5698	0.9049
ecoli-0-1_vs_2-3-5	0.8243	0.8738	0.7129	0.6248	0.8326	0.7828	0.9044
ecoli-0-1_vs_5	0.8421	0.8858	0.8021	0.7276	0.8421	0.8126	0.9047
ecoli-0-2-3-4_vs_5	0.8502	0.8834	0.8001	0.6638	0.8482	0.7898	0.8881
ecoli-0-2-6-7_vs_3-5	0.8392	0.8971	0.7771	0.1894	0.8436	0.7322	0.9222
ecoli-0-3-4-6_vs_5	0.8516	0.8864	0.7814	0.6955	0.8295	0.8276	0.8995
ecoli-0-3-4-7_vs_5-6	0.8476	0.8884	0.7438	0.3044	0.8559	0.6719	0.8985
ecoli-0-3-4_vs_5	0.8493	0.8834	0.8062	0.7223	0.8493	0.8196	0.8888
ecoli-0-4-6_vs_5	0.8382	0.8801	0.7916	0.6146	0.8437	0.8145	0.8887
ecoli-0-6-7_vs_3-5	0.8468	0.8971	0.7792	0.1894	0.8228	0.6228	0.9254
ecoli-0-6-7_vs_5	0.8259	0.8505	0.7588	0.3414	0.8003	0.7707	0.8771
ecoli-0_vs_1	0.9516	0.9313	0.9213	0.7850	0.9618	0.9313	0.9408
ecoli1	0.8032	0.8715	0.6072	0.2597	0.8348	0.8397	0.8556
ecoli2	0.8955	0.8690	0.3216	0.5343	0.9414	0.9330	0.8814
ecoli3	0.7461	0.8721	0.7674	0.0000	0.7462	0.8150	0.8615
ecoli4	0.8235	0.9102	0.8358	0.5865	0.8594	0.8898	0.9192
glass-0-1-2-3_vs_4-5-6	0.8757	0.8085	0.9227	0.6575	0.8749	0.7952	0.9269
glass-0-1-4-6_vs_2	0.1945	0.4824	0.4476	0.0000	0.0000	0.0000	0.5046
glass-0-1-5_vs_2	0.4222	0.6070	0.2686	0.0000	0.2155	0.0000	0.4394
glass-0-1-6_vs_2	0.3065	0.4965	0.4172	0.0000	0.2155	0.0000	0.4036
glass-0-1-6_vs_5	0.8750	0.9065	0.8938	0.1414	0.7616	0.4808	0.8054
glass-0-4_vs_5	0.8198	0.9552	0.9361	0.3414	0.6828	0.6741	0.9004
glass-0-6_vs_5	0.9364	0.9041	0.8751	0.1414	0.8243	0.5621	0.9364
glass0	0.8020	0.5960	0.5259	0.0000	0.8232	0.7667	0.5508
glass1	0.7387	0.5268	0.0000	0.0000	0.7294	0.2247	0.3839
glass2	0.2343	0.5234	0.4839	0.0000	0.0000	0.0000	0.5905
glass4	0.6572	0.7452	0.8752	0.2309	0.6698	0.5667	0.8357
glass5	0.5975	0.9020	0.8718	0.1414	0.5414	0.1414	0.6640
glass6	0.8205	0.8151	0.8412	0.7605	0.8227	0.8329	0.8393
habermanImb	0.5012	0.5088	0.0000	0.0000	0.4315	0.2176	0.0000
ionosphere-bredB_vs_g	0.3398	0.1366	0.5075	0.2000	0.2000	0.3414	0.3414
ionosphere-bred_vs_g	0.6227	0.4243	0.6211	0.6243	0.6243	0.6243	0.6243
iris0	1,0000	1,0000	1,0000	0.9897	1,0000	1,0000	1,0000
kddcup-buffer_overflow_vs_back	0.9266	0.9202	0.8847	0.4756	0.8328	0.8328	0.9259
kddcup-guess_passwd_vs_satan	0.9904	0.9898	0.9885	0.9811	0.9904	0.9904	0.9898
kddcup-land_vs_portsweep	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
kddcup-land_vs_satan	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997
kddcup-rootkit-imap_vs_back	0.9196	0.9718	0.9572	0.8985	0.9194	0.9458	0.9456
magic-hredB_vs_gredB	0.5042	0.6938	0.4203	0.0000	0.3869	0.0894	0.6358
magic-hred_vs_gred	0.4753	0.7012	0.3212	0.0000	0.4253	0.3412	0.7706
movement_libras-1	0.8321	0.9519	0.7993	0.0000	0.7316	0.3159	0.9502
new-thyroid1	0.9621	0.8540	0.9247	0.6134	0.9368	0.7494	0.9859
new-thyroid2	0.9647	0.8540	0.9275	0.6336	0.9325	0.7494	0.9888
page-blocks-1-3_vs_4	0.8668	0.7677	0.5409	0.4020	0.8513	0.6601	0.9371
page-blocks0	0.8547	0.7249	0.0773	0.0000	0.8411	0.5708	0.6788
phoneme-1redB_vs_0redB	0.2378	0.6826	0.3649	0.0000	0.0000	0.0000	0.7076
phoneme-1red_vs_0red	0.4011	0.7419	0.3285	0.0000	0.1837	0.0000	0.7569
segment-5red_vs_1-2-3	0.6521	0.7753	0.5772	0.0000	0.4888	0.1000	0.8400
segment-6red_vs_3-4-5	0.7868	0.8275	0.7412	0.0000	0.4461	0.4461	0.9341
segment-7red_vs_2-4-5-6	1,0000	0.9866	0.9839	0.9633	1,0000	1,0000	0.9996
segment0	0.9811	0.5264	0.4061	0.0000	0.9811	0.8957	0.7571
shuttle-2_vs_1red	0.9680	0.8426	0.5695	0.6949	0.9572	0.7074	0.9554

shuttle-2_vs_5	1,0000	0.9248	0.9576	0.8426	1,0000	0.8548	0.9997
shuttle-6-7_vs_1redB	0.9732	0.6647	0.9624	0.3944	0.9310	0.3944	0.9732
shuttle-6_vs_2-3	0.8828	0.8028	0.8198	0.6243	0.8828	0.6243	0.8828
shuttle-c0-vs-c4	0.9960	0.9930	0.9950	0.9960	0.9960	0.9960	0.9997
shuttle-c2-vs-c4	0.9414	0.7414	1,0000	0.7414	0.9414	0.7414	0.9414
texture-12red_vs_13-14	0.9995	0.9894	0.9844	0.9633	1,0000	0.9995	0.9995
texture-2red_vs_3-4	0.8706	0.8065	0.2018	0.0667	0.8304	0.7243	0.9795
texture-6red_vs_7-8	1,0000	0.8870	0.7501	0.9253	0.9732	0.9732	1,0000
texture-7red_vs_2-3-4-6	1,0000	0.9391	0.8787	1,0000	1,0000	1,0000	1,0000
vehicle0	0.8549	0.7087	0.2608	0.0000	0.8641	0.3662	0.5377
vehicle1	0.6110	0.6410	0.0000	0.0000	0.5985	0.0000	0.0356
vehicle2	0.9119	0.7336	0.0000	0.0000	0.9218	0.4648	0.1974
vehicle3	0.6625	0.6556	0.0000	0.0000	0.6431	0.0436	0.0783
wdbc-MredB_vs_B	0.5456	0.8618	0.9058	0.0000	0.5575	0.5575	0.7491
wdbc-Mred_vs_B	0.8828	0.8722	0.9701	0.5414	0.7414	0.7414	0.7400
winequality-red-3_vs_5	0.0000	0.4100	0.7325	0.0000	0.0000	0.0000	0.4227
winequality-red-4	0.2693	0.6475	0.0000	0.0000	0.0000	0.0000	0.6762
winequality-red-8_vs_6	0.3391	0.8163	0.6300	0.0000	0.0000	0.0000	0.7360
winequality-red-8_vs_6-7	0.1982	0.7260	0.5588	0.0000	0.0000	0.0000	0.5027
winequality-white-3-9_vs_5	0.2679	0.5381	0.3705	0.0000	0.0894	0.0000	0.5154
winequality-white-3_vs_7	0.1410	0.4952	0.4902	0.0000	0.1000	0.0000	0.4815
winequality-white-9_vs_4	0.7907	0.7322	0.7222	0.4000	0.6000	0.8000	0.7969
wisconsinImb	0.9495	0.9540	0.9287	0.3757	0.9504	0.9475	0.9452
yeast-0-2-5-6_vs_3-7-8-9	0.7304	0.8011	0.0000	0.0000	0.7223	0.5152	0.6063
yeast-0-2-5-7-9_vs_3-6-8	0.8681	0.9109	0.2112	0.0000	0.8927	0.8600	0.8459
yeast-0-3-5-9_vs_7-8	0.6319	0.6730	0.1098	0.0000	0.5728	0.4595	0.6821
yeast-0-5-6-7-9_vs_4	0.5908	0.8098	0.4221	0.0000	0.5665	0.4176	0.7602
yeast-1-2-8-9_vs_7	0.3535	0.6697	0.2393	0.0000	0.2445	0.0000	0.6319
yeast-1-4-5-8_vs_7	0.3519	0.5904	0.2428	0.0000	0.0000	0.0000	0.6405
yeast-1_vs_7	0.5162	0.6604	0.3598	0.0000	0.2449	0.0000	0.7446
yeast-2_vs_4	0.7898	0.8695	0.5613	0.0000	0.7948	0.6872	0.8836
yeast-2_vs_8	0.6922	0.7283	0.4234	0.6965	0.7283	0.7283	0.7274
yeast1	0.6169	0.6659	0.1154	0.0000	0.6121	0.4127	0.1191
yeast3	0.8142	0.8696	0.2102	0.0000	0.8384	0.7079	0.7293
yeast4	0.5969	0.8346	0.3595	0.0000	0.5173	0.3855	0.8315
yeast5	0.9094	0.9402	0.8583	0.1374	0.8738	0.9044	0.9603
yeast6	0.7041	0.8607	0.5309	0.0756	0.6693	0.8175	0.8937
Mean	0.7115	0.7856	0.5988	0.3107	0.6518	0.5405	0.7695

ANEXO G

G-Mean obtenido por los algoritmos del estado del arte seleccionados para comparar con las variantes IFROWANN

En este apéndice se presentan los resultados experimentales de los algoritmos del estado del arte selecionados para comparar con los algoritmos IFROWANN introducidos en el Capítulo 4.

En la tabla G.1 se muestran los resultados de G-mean para los algoritmos de preprocesamiento seleccionados usando como clasificadores C4.5, kNN y SVM. Las columnas de las tablas se corresponden con:

Tabla G.1: Resultados de G-mean para los algoritmos de preprocesamiento

Datasets	**S-kNN**	**S-C4.5**	**S-SVM**	**SB-kNN**	**SB-C4.5**	**SB-SVM**	**SE-kNN**	**SE-C4.5**	**SE-SVM**
abalone-17_vs_7-8-9-10	0.8298	0.6983	0.8504	0.8297	0.7766	0.8495	0.8297	0.7025	0.8422
abalone-19_vs_10-11-12-13	0.5885	0.3103	0.6787	0.5678	0.4277	0.7101	0.5660	0.3836	0.7137
abalone-20_vs_8-9-10	0.7879	0.6124	0.9012	0.8107	0.6718	0.9018	0.7755	0.7506	0.9040
abalone-21_vs_8	0.8343	0.8580	0.8313	0.8318	0.7777	0.8305	0.8330	0.8568	0.8353
abalone-3_vs_11	0.9938	0.9979	0.9979	0.9938	0.9958	0.9979	0.9927	0.9958	0.9969
abalone19	0.5832	0.3389	0.7795	0.5851	0.2337	0.7652	0.5804	0.4175	0.7797
abalone9-18	0.7687	0.7698	0.8267	0.7825	0.7546	0.8298	0.7890	0.7443	0.8336
appendicitisImb	0.7814	0.6652	0.7866	0.7704	0.7317	0.7873	0.7068	0.7004	0.7319
cleveland-0_vs_4	0.8688	0.7186	0.8749	0.9202	0.6938	0.9046	0.8728	0.7516	0.9103
cleveland-4	0.7581	0.1376	0.7796	0.7207	0.1298	0.7920	0.7614	0.3636	0.8203

ecoli-0-1-3-7_vs_2-6	0.6878	0.3237	0.7567	0.7232	0.7100	0.7365	0.6818	0.5144	0.7586
ecoli-0-1-4-6_vs_5	0.8754	0.8374	0.8728	0.8806	0.8448	0.8748	0.8747	0.7867	0.8707
ecoli-0-1-4-7_vs_2-3-5-6	0.8756	0.8840	0.8768	0.8390	0.8554	0.7899	0.8732	0.8785	0.8882
ecoli-0-1-4-7_vs_5-6	0.8919	0.8544	0.8841	0.8760	0.7516	0.8925	0.8788	0.8088	0.8878
ecoli-0-1_vs_2-3-5	0.8472	0.8369	0.8638	0.7704	0.7261	0.7894	0.8822	0.8145	0.8704
ecoli-0-1_vs_5	0.8964	0.8386	0.8853	0.8977	0.8553	0.8871	0.8893	0.8497	0.8742
ecoli-0-2-3-4_vs_5	0.8824	0.8534	0.8803	0.8813	0.8828	0.8803	0.8897	0.8813	0.8811
ecoli-0-2-6-7_vs_3-5	0.8638	0.7789	0.8838	0.8439	0.7950	0.8857	0.8847	0.7762	0.8569
ecoli-0-3-4-6_vs_5	0.8833	0.8945	0.8905	0.8803	0.8470	0.8877	0.8887	0.9047	0.8877
ecoli-0-3-4-7_vs_5-6	0.8892	0.8212	0.8984	0.8979	0.8523	0.9077	0.8989	0.8345	0.9050
ecoli-0-3-4_vs_5	0.8846	0.8957	0.8824	0.8847	0.8908	0.8824	0.8888	0.9042	0.8824
ecoli-0-4-6_vs_5	0.8839	0.8906	0.8801	0.8876	0.8685	0.8892	0.8791	0.8996	0.8723
ecoli-0-6-7_vs_3-5	0.8575	0.8306	0.8164	0.8814	0.8113	0.8257	0.8736	0.7980	0.8191
ecoli-0-6-7_vs_5	0.8546	0.7701	0.8680	0.8705	0.7895	0.8681	0.8392	0.6999	0.8355
ecoli-0_vs_1	0.9618	0.9831	0.9761	0.9618	0.9505	0.9726	0.9491	0.9831	0.9691
ecoli1	0.9001	0.8902	0.8923	0.9001	0.8927	0.8923	0.8877	0.8695	0.8971
ecoli2	0.9159	0.8920	0.9148	0.9141	0.9059	0.9149	0.9086	0.8756	0.9073
ecoli3	0.8752	0.8557	0.8940	0.8774	0.8764	0.8886	0.8805	0.8316	0.8853
ecoli4	0.9596	0.9502	0.9238	0.9546	0.8935	0.9500	0.9512	0.9205	0.9192
glass-0-1-2-3_vs_4-5-6	0.9180	0.8799	0.9001	0.9192	0.8835	0.8984	0.9281	0.9139	0.8998
glass-0-1-4-6_vs_2	0.6698	0.4649	0.6591	0.6333	0.5975	0.6565	0.6509	0.7700	0.6892
glass-0-1-5_vs_2	0.6643	0.6863	0.3845	0.6299	0.4305	0.3784	0.6688	0.4818	0.4223
glass-0-1-6_vs_2	0.6853	0.6721	0.4743	0.6968	0.5408	0.4941	0.6998	0.5076	0.4960
glass-0-1-6_vs_5	0.9348	0.9560	0.9156	0.9409	0.9141	0.9188	0.9379	0.9102	0.9126
glass-0-4_vs_5	0.9155	0.9688	0.9552	0.7414	0.9871	0.8584	0.9216	0.9813	0.9482
glass-0-6_vs_5	0.9248	0.9477	0.9410	0.9796	0.9239	0.8822	0.9692	0.9795	0.9410
glass0	0.7823	0.7716	0.7086	0.7962	0.8022	0.7204	0.7764	0.8001	0.7231
glass1	0.7337	0.6782	0.5263	0.7381	0.7590	0.5516	0.7567	0.7109	0.6177
glass2	0.7095	0.5757	0.5794	0.6855	0.6915	0.5647	0.7100	0.6728	0.5232
glass4	0.9302	0.7615	0.8859	0.8178	0.8423	0.8562	0.9357	0.9072	0.8750
glass5	0.9366	0.9777	0.8750	0.9363	0.9700	0.9285	0.9281	0.9190	0.9262
glass6	0.8752	0.8888	0.9191	0.8593	0.8650	0.8989	0.8620	0.9135	0.9134
habermanImb	0.5707	0.6186	0.6394	0.5436	0.6405	0.6172	0.5226	0.6374	0.6335
ionosphere-bredB_vs_g	0.9215	0.5910	0.9152	0.3414	0.3978	0.5392	0.9238	0.7843	0.9286
ionosphere-bred_vs_g	0.8518	0.5399	0.6553	0.1414	0.6813	0.6227	0.8659	0.5220	0.7209
iris0	1,0000	0.9897	1,0000	1,0000	0.9897	1,0000	1,0000	0.9897	1,0000
kddcup-buffer_overflow_vs_back	0.9995	0.9826	1,0000	0.9995	0.9826	1,0000	0.9995	0.9826	1,0000
kddcup-guess_passwd_vs_satan	0.9907	0.9994	1,0000	0.9907	0.9994	1,0000	0.9907	0.9994	1,0000
kddcup-land_vs_portsweep	0.9995	1,0000	0.9995	0.9995	1,0000	0.9995	0.9995	1,0000	0.9995
kddcup-land_vs_satan	0.9984	1,0000	0.9402	0.9984	1,0000	0.9991	0.9994	1,0000	1,0000
kddcup-rootkit-imap_vs_back	0.9995	0.9995	0.9998	0.9993	0.9995	0.9998	0.9787	0.9995	1,0000
magic-hredB_vs_gredB	0.7317	0.7455	0.8312	0.6681	0.7176	0.8308	0.6799	0.7093	0.8342
magic-hred_vs_gred	0.7823	0.6705	0.7754	0.7611	0.6275	0.7869	0.7905	0.7484	0.7934
movement_libras-1	0.9640	0.8427	0.9053	0.9624	0.8132	0.9057	0.9624	0.8219	0.9275
new-thyroid1	0.9859	0.9368	0.9859	0.8393	0.9421	0.9739	0.9831	0.9507	0.9831
new-thyroid2	0.9803	0.9258	0.9803	0.8877	0.9479	0.9767	0.9485	0.9162	0.9775
page-blocks-1-3_vs_4	0.9408	0.9887	0.7976	0.9175	0.9875	0.6909	0.9311	0.9864	0.7766
page-blocks0	0.9239	0.9385	0.8712	0.9186	0.9445	0.8684	0.9189	0.9375	0.8693
phoneme-1redB_vs_0redB	0.5178	0.4939	0.7145	0.4655	0.4076	0.7155	0.5345	0.4154	0.7185
phoneme-1red_vs_0red	0.7417	0.6742	0.7536	0.7522	0.6418	0.7535	0.7558	0.5702	0.7573
segment-5red_vs_1-2-3	0.8422	0.7396	0.8281	0.8423	0.7715	0.8269	0.8607	0.8247	0.8248
segment-6red_vs_3-4-5	0.9506	0.9883	0.9733	0.9485	0.9847	0.9733	0.9570	0.9862	0.9728
segment-7red_vs_2-4-5-6	0.9985	0.9989	0.9989	0.9973	0.9989	0.9985	0.9996	0.9996	0.9996

segment0	0.9891	0.9878	0.9921	0.9886	0.9962	0.9926	0.9903	0.9904	0.9919
shuttle-2_vs_1red	0.9881	0.9994	0.9078	0.9881	0.9994	0.9074	0.9783	0.9996	0.8952
shuttle-2_vs_5	0.9977	0.9991	0.9959	0.9985	0.9994	0.9959	1,0000	0.9994	0.9971
shuttle-6-7_vs_1redB	1,0000	0.9789	0.8448	1,0000	0.9789	0.8711	1,0000	0.9789	0.8132
shuttle-6_vs_2-3	0.8779	0.9977	0.9368	0.9368	0.9977	0.9345	0.9391	0.9977	0.9391
shuttle-c0-vs-c4	0.9960	0.9994	0.9960	0.9960	1,0000	0.9960	0.9960	0.9997	0.9960
shuttle-c2-vs-c4	1,0000	0.9958	0.9960	1,0000	0.9958	0.9960	1,0000	1,0000	0.9960
texture-12red_vs_13-14	0.9985	0.9975	0.9985	0.9985	0.9980	0.9985	0.9990	0.9995	0.9995
texture-2red_vs_3-4	0.9393	0.8853	0.9380	0.9644	0.8725	0.9401	0.9451	0.8972	0.9305
texture-6red_vs_7-8	0.9980	0.9172	0.9975	0.9985	0.9168	0.9985	0.9995	0.9187	0.9985
texture-7red_vs_2-3-4-6	1,0000	0.9146	1,0000	1,0000	0.9146	1,0000	1,0000	0.9146	1,0000
vehicle0	0.9057	0.9359	0.9595	0.9041	0.9386	0.9612	0.9106	0.9280	0.9573
vehicle1	0.7269	0.6908	0.7439	0.7353	0.7006	0.7496	0.7421	0.7529	0.7349
vehicle2	0.9017	0.9586	0.9168	0.9020	0.9547	0.9198	0.8969	0.9521	0.9189
vehicle3	0.7395	0.7071	0.6975	0.7424	0.6917	0.7157	0.7212	0.7154	0.7147
wdbc-MredB_vs_B	0.8670	0.8202	0.9274	0.9329	0.7792	0.9256	0.9069	0.7975	0.9360
wdbc-Mred_vs_B	0.9138	0.7285	0.9249	0.8727	0.8766	0.9916	0.9287	0.7915	0.9263
winequality-red-3_vs_5	0.7150	0.2771	0.6105	0.7099	0.1367	0.6597	0.7495	0.1362	0.7131
winequality-red-4	0.5624	0.4180	0.6898	0.5787	0.4894	0.6747	0.5425	0.4024	0.6877
winequality-red-8_vs_6	0.7069	0.5976	0.8251	0.7177	0.5480	0.8184	0.6603	0.7374	0.8135
winequality-red-8_vs_6-7	0.6248	0.5481	0.8067	0.6260	0.2609	0.7765	0.6443	0.5619	0.7258
winequality-white-3-9_vs_5	0.6678	0.4151	0.6615	0.6623	0.2919	0.6360	0.6576	0.4155	0.6618
winequality-white-3_vs_7	0.7183	0.4697	0.6701	0.7807	0.4598	0.6678	0.7053	0.4620	0.5985
winequality-white-9_vs_4	0.8049	0.3904	0.6905	0.7945	0.3811	0.6869	0.8056	0.5549	0.6945
wisconsinImb	0.9747	0.9600	0.9718	0.9677	0.9479	0.9645	0.9674	0.9394	0.9777
yeast-0-2-5-6_vs_3-7-8-9	0.7924	0.7283	0.7961	0.7842	0.7742	0.7895	0.7862	0.7388	0.7896
yeast-0-2-5-7-9_vs_3-6-8	0.8982	0.8979	0.9067	0.9040	0.8895	0.9067	0.9015	0.9094	0.9062
yeast-0-3-5-9_vs_7-8	0.7216	0.7130	0.7361	0.7322	0.6924	0.7340	0.7368	0.6760	0.7382
yeast-0-5-6-7-9_vs_4	0.7867	0.8208	0.7812	0.7778	0.7551	0.7824	0.7967	0.7295	0.7827
yeast-1-2-8-9_vs_7	0.6539	0.6357	0.7209	0.6522	0.4749	0.7194	0.6728	0.5379	0.7236
yeast-1-4-5-8_vs_7	0.6740	0.4604	0.6518	0.6733	0.3640	0.6493	0.6702	0.4524	0.6562
yeast-1_vs_7	0.7007	0.6742	0.7334	0.7057	0.6269	0.7649	0.7133	0.6058	0.7387
yeast-2_vs_4	0.9092	0.8926	0.8817	0.9090	0.9046	0.8930	0.9256	0.8780	0.8816
yeast-2_vs_8	0.7984	0.7916	0.7260	0.7939	0.7456	0.7244	0.7918	0.8129	0.7957
yeast1	0.6989	0.7071	0.7056	0.6990	0.6966	0.7051	0.7047	0.7064	0.7112
yeast3	0.9013	0.9033	0.9041	0.9005	0.8870	0.9075	0.8992	0.9069	0.9095
yeast4	0.8267	0.7270	0.8069	0.8098	0.7154	0.8169	0.8166	0.7695	0.8172
yeast5	0.9707	0.9401	0.9714	0.9704	0.9503	0.9711	0.9682	0.9720	0.9714
yeast6	0.8601	0.8134	0.8643	0.8580	0.8326	0.8804	0.8575	0.8335	0.8640
Mean	0.8496	0.7860	0.8453	0.8309	0.7776	0.8401	0.8498	0.7953	0.8485

La tabla G.2 muestra los resultados de AUC en cuanto a test para los algoritmos basados en aprendizaje sensible al coste (CS-C4.5 y CS-SVM), así como el algoritmo ensemble EUSBOOST.

Tabla G.2: Resultados de G-mean para los algoritmos cost sensitive y el ensemble EUSBOOST

Datasets	**CS-C4.5**	**CS-SVM**	**EUSBOOST**
abalone-17_vs_7-8-9-10	0.6338	0.0000	0.7647

abalone-19_vs_10-11-12-13	0.2614	0.0000	0.5939
abalone-20_vs_8-9-10	0.5734	0.0000	0.7088
abalone-21_vs_8	0.8433	0.0000	0.8118
abalone-3_vs_11	0.9623	0.4309	0.9623
abalone19	0.3071	0.7557	0.7066
abalone9-18	0.5251	0.8718	0.7445
appendicitisImb	0.5579	0.0000	0.7591
cleveland-0_vs_4	0.5569	0.6163	0.7616
cleveland-4	0.6568	0.6804	0.7458
ecoli-0-1-3-7_vs_2-6	0.7346	0.7414	0.7030
ecoli-0-1-4-6_vs_5	0.8236	0.7500	0.8838
ecoli-0-1-4-7_vs_2-3-5-6	0.8708	0.6643	0.9056
ecoli-0-1-4-7_vs_5-6	0.8436	0.7678	0.8953
ecoli-0-1_vs_2-3-5	0.7135	0.7189	0.8168
ecoli-0-1_vs_5	0.8039	0.7502	0.9181
ecoli-0-2-3-4_vs_5	0.8168	0.8274	0.8730
ecoli-0-2-6-7_vs_3-5	0.8371	0.7368	0.8258
ecoli-0-3-4-6_vs_5	0.8384	0.8877	0.8805
ecoli-0-3-4-7_vs_5-6	0.7352	0.7885	0.8935
ecoli-0-3-4_vs_5	0.9221	0.8480	0.9167
ecoli-0-4-6_vs_5	0.8178	0.8564	0.9027
ecoli-0-6-7_vs_3-5	0.8537	0.6764	0.8272
ecoli-0-6-7_vs_5	0.8715	0.6844	0.8936
ecoli-0_vs_1	0.9831	0.9658	0.9652
ecoli1	0.9109	0.9022	0.8855
ecoli2	0.8870	0.0000	0.9175
ecoli3	0.8278	0.6888	0.8793
ecoli4	0.8534	0.9512	0.8991
glass-0-1-2-3_vs_4-5-6	0.8746	0.8294	0.9075
glass-0-1-4-6_vs_2	0.5646	0.0000	0.7826
glass-0-1-5_vs_2	0.3946	0.0000	0.7745
glass-0-1-6_vs_2	0.4726	0.0000	0.7627
glass-0-1-6_vs_5	0.9885	0.0000	0.9885
glass-0-4_vs_5	0.9940	0.8828	0.9940
glass-0-6_vs_5	0.9949	0.4243	0.9949
glass0	0.8136	0.0729	0.8180
glass1	0.7143	0.5986	0.7901
glass2	0.5259	0.4868	0.7509
glass4	0.8344	0.9046	0.8632
glass5	0.9340	0.9724	0.9876
glass6	0.8806	0.8628	0.9136
habermanImb	0.5004	0.1310	0.6258
ionosphere-bredB_vs_g	0.0000	0.1414	0.3596
ionosphere-bred_vs_g	0.3382	0.4243	0.7309
iris0	0.9897	1,0000	0.9897
kddcup-buffer_overflow_vs_back	0.9826	1,0000	0.9826
kddcup-guess_passwd_vs_satan	0.9991	0.9997	0.9876
kddcup-land_vs_portsweep	1,0000	0.9727	0.9785
kddcup-land_vs_satan	1,0000	0.9729	0.9783
kddcup-rootkit-imap_vs_back	0.9789	1,0000	0.9789
magic-hredB_vs_gredB	0.7408	0.0000	0.7692
magic-hred_vs_gred	0.7199	0.0000	0.7648
movement_libras-1	0.8367	0.8168	0.8949

new-thyroid1	0.9741	0.9675	0.9567
new-thyroid2	0.9796	0.9824	0.9596
page-blocks-1-3_vs_4	0.9779	0.8419	0.9864
page-blocks0	0.9454	0.9251	0.9543
phoneme-1redB_vs_0redB	0.4238	0.0000	0.7127
phoneme-1red_vs_0red	0.5482	0.0000	0.6878
segment-5red_vs_1-2-3	0.8625	0.0000	0.8746
segment-6red_vs_3-4-5	1,0000	0.9995	0.9949
segment-7red_vs_2-4-5-6	1,0000	1,0000	1,0000
segment0	0.9919	0.9965	0.9929
shuttle-2_vs_1red	0.9997	0.9196	0.9994
shuttle-2_vs_5	1,0000	0.9995	1,0000
shuttle-6-7_vs_1redB	0.9761	0.5228	0.9997
shuttle-6_vs_2-3	0.9977	1,0000	0.9908
shuttle-c0-vs-c4	0.9997	1,0000	1,0000
shuttle-c2-vs-c4	1,0000	1,0000	1,0000
texture-12red_vs_13-14	0.9266	1,0000	0.8788
texture-2red_vs_3-4	0.8760	0.0000	0.9098
texture-6red_vs_7-8	0.9397	0.7667	0.9410
texture-7red_vs_2-3-4-6	0.9460	1,0000	0.9480
vehicle0	0.9285	0.9485	0.9530
vehicle1	0.6958	0.7059	0.7525
vehicle2	0.9431	0.9570	0.9757
vehicle3	0.7226	0.7867	0.7767
wdbc-MredB_vs_B	0.8161	0.6899	0.8928
wdbc-Mred_vs_B	0.8677	0.7372	0.9155
winequality-red-3_vs_5	0.5970	0.7156	0.7437
winequality-red-4	0.4714	0.6624	0.6447
winequality-red-8_vs_6	0.4867	0.7411	0.7365
winequality-red-8_vs_6-7	0.5504	0.7498	0.6554
winequality-white-3-9_vs_5	0.5441	0.6702	0.6604
winequality-white-3_vs_7	0.4258	0.5633	0.7542
winequality-white-9_vs_4	0.5840	0.5843	0.7453
wisconsinImb	0.9635	0.9718	0.9594
yeast-0-2-5-6_vs_3-7-8-9	0.7719	0.0000	0.7898
yeast-0-2-5-7-9_vs_3-6-8	0.8973	0.0000	0.8992
yeast-0-3-5-9_vs_7-8	0.6502	0.0000	0.6337
yeast-0-5-6-7-9_vs_4	0.6946	0.0000	0.7713
yeast-1-2-8-9_vs_7	0.6206	0.0000	0.7434
yeast-1-4-5-8_vs_7	0.3757	0.0000	0.4766
yeast-1_vs_7	0.4652	0.0000	0.7459
yeast-2_vs_4	0.8839	0.0000	0.9580
yeast-2_vs_8	0.8470	0.7219	0.7379
yeast1	0.6642	0.5747	0.7107
yeast3	0.9112	0.8948	0.9299
yeast4	0.6824	0.8133	0.8348
yeast5	0.9310	0.9650	0.9453
yeast6	0.7845	0.8730	0.8855
Mean	0.7749	0.6010	0.8493

ANEXO H

Resultados de G-mean sumarizados atendiendo al IR para los algoritmos IFROWANN

Tabla H.1: G-mean promediada por intervalos de IR

Method	all	<9	>9	>33
TL-$\mathscr{W}_1$	0.7858	0.8267	0.7745	0.7435
TL-$\mathscr{W}_2$	0.6541	0.6766	0.6480	0.6445
TL-$\mathscr{W}_3$	0.4622	0.4812	0.4570	0.4136
TL-$\mathscr{W}_4$	0.6864	0.8335	0.6460	0.5535
TL-$\mathscr{W}_5$	0.6309	0.7641	0.5943	0.5106
TL-$\mathscr{W}_6$	0.7886	0.7621	0.7958	0.7636
AV-$\mathscr{W}_1$	0.7993	0.7781	0.8051	0.7712
AV-$\mathscr{W}_2$	0.6093	0.5013	0.6389	0.6633
AV-$\mathscr{W}_3$	0.3084	0.3079	0.3085	0.3202
AV-$\mathscr{W}_4$	0.6612	0.8356	0.6132	0.5150
AV-$\mathscr{W}_5$	0.5719	0.6630	0.5468	0.4625
AV-$\mathscr{W}_6$	0.7916	0.6806	0.8221	0.7871
MIN-$\mathscr{W}_1$	0.7856	0.7601	0.7926	0.7706
MIN-$\mathscr{W}_2$	0.5988	0.4662	0.6353	0.6497
MIN-$\mathscr{W}_3$	0.3107	0.2639	0.3235	0.3575
MIN-$\mathscr{W}_4$	0.6518	0.8187	0.6059	0.5287
MIN-$\mathscr{W}_5$	0.5405	0.6348	0.5146	0.4574
MIN-$\mathscr{W}_6$	0.7695	0.6381	0.8056	0.7942
VCCAD-MIN	0.7115	0.8209	0.6814	0.6182
VCCAD-AV	0.7192	0.8287	0.6891	0.6125
VCCAD-TL	0.7280	0.8249	0.7014	0.6187

ANEXO I

Resultados de G-mean sumarizados atendiendo al IR para los algoritmos del estado del arte

Tabla I.1: G-mean obtenida por los algoritmos del estado del arte

Method	all	<9	>9	>33
SMOTE-KNN	0.8496	0.8656	0.8452	0.8382
SMOTE-C4.5	0.7860	0.8529	0.7676	0.7317
SMOTE-SVM	0.8453	0.8585	0.8417	0.8592
SMOTE-ENN-kNN	0.8498	0.8573	0.8478	0.8363
SMOTE-ENN-C4.5	0.7953	0.8601	0.7775	0.7547
SMOTE-ENN-SVM	0.8485	0.8602	0.8452	0.8603
C4.5-CS	0.7749	0.8482	0.7547	0.7308
SVM-CS	0.6010	0.7165	0.5693	0.5756
EUSBOOST	0.8493	0.8806	0,.8407	0.8222
SMOTE-RSB*-kNN	0.8309	0.8530	0.8248	0.8172
SMOTE-RSB*-C4.5	0.7776	0.8611	0.7546	0.7207
SMOTE-RSB*-SVM	0.8401	0.8581	0,.8352	0.8522

ANEXO J

Análisis gráfico del comportamiento de la media geométrica (G-mean)

Figure J.1: G-mean para todos los conjuntos de datos ordenados atendiendo a su IR, para nuestra mejor propuesta (AV-$\mathscr{W}_6$) y los algoritmos más competitivos del estado del arte (SMOTE-RSB$_*$-kNN y EUSBOOST)

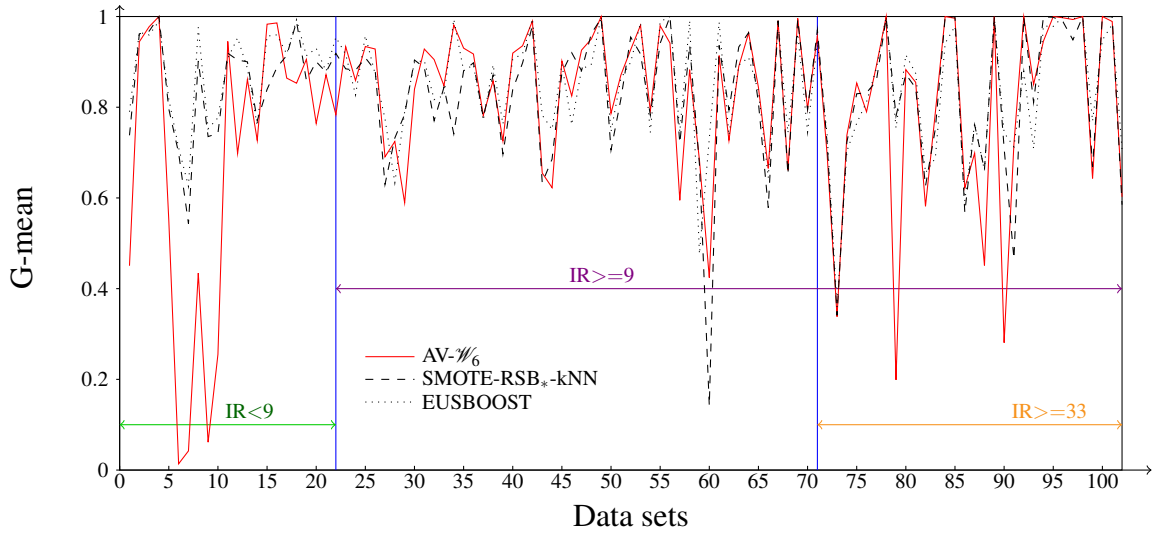
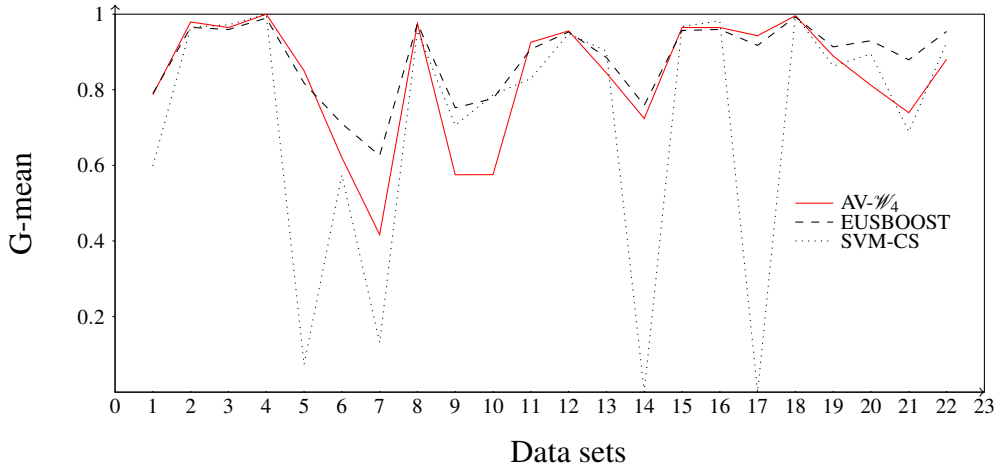
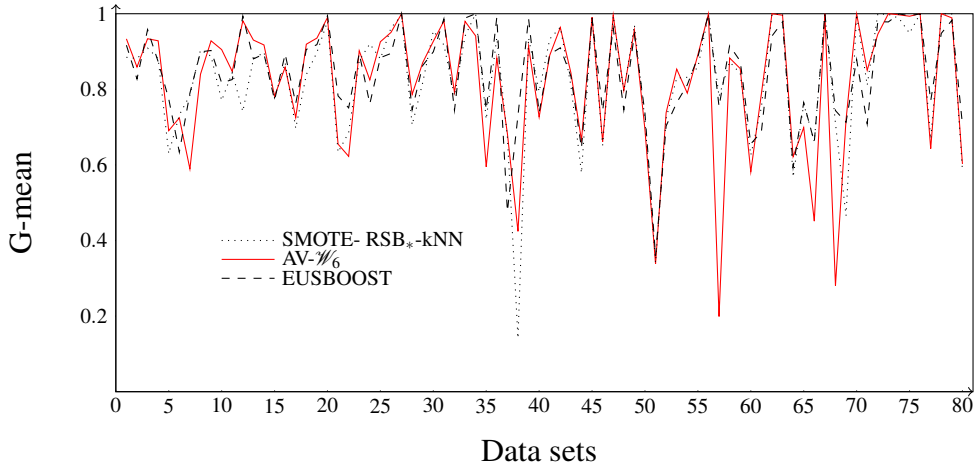


Figure J.2: G-mean para todos los bloques de datos. Los conjuntos están ordenados atendiendo a su IR.

(a) **IR<9** Low Imbalance



(b) **IR>=9** High Imbalance



(c) **IR>=33** Very High Imbalance