UNIVERSIDAD DE GRANADA

DOCTORADO EN TECNOLOGÍAS DE LA INFORMACIÓN Y LA COMUNICACIÓN

RETOS EN CLASIFICACIÓN ORDINAL: REDES NEURONALES ARTIFICIALES Y MÉTODOS BASADOS EN PROYECCIONES

CHALLENGES IN ORDINAL CLASSIFICATION: ARTIFICIAL NEURAL NETWORKS AND PROJECTION-BASED METHODS

Doctor Internacional
International Doctorate

Doctorando:

JAVIER SÁNCHEZ MONEDERO

Directores:

CÉSAR HERVÁS MARTÍNEZ

PEDRO ANTONIO GUTIÉRREZ PEÑA

Septiembre, Curso Académico 2012–2013

Editor: Editorial de la Universidad de Granada
Autor: Javier Sánchez Monedero
D.L.: GR 494-2014
ISBN: 978-84-9028-800-9

AUTHOR:
Javier Sánchez Monedero: *Retos en clasificación ordinal: redes neuronales artificiales y métodos basados en proyecciones, Challenges in ordinal classification: artificial neural networks and projection-based methods* , © September 2013

Contact: jsanchezm at uco.es
Learning and Artificial Neural Networks (AYRNA) research group
Department of Computer Science and Numerical Analysis
University of Córdoba
Códoba (Spain)
http://www.uco.es/ayrna/

SUPERVISORS:
César Hervás Martínez
Pedro Antonio Gutiérrez Peña

DECLARACIÓN

La memoria titulada «Retos en clasificación ordinal: redes neuronales artificiales y métodos basados en proyecciones », que presenta D. Javier Sánchez Monedero para optar al grado de Doctor, ha sido realizada dentro del *Doctorado en Tecnologías de la Información y la Comunicación* de la Universidad de Granada bajo la dirección del Catedrático de Universidad D. César Hervás Martínez y del profesor D. Pedro Antonio Gutiérrez Peña.

El doctorando D. Javier Sánchez Monedero y los directores de la tesis D. César Hervás Martínez y D. Pedro Antonio Gutiérrez Peña garantizamos, al firmar esta Tesis Doctoral, que el trabajo ha sido realizado por el doctorando, bajo la dirección de los directores de la Tesis y hasta donde nuestro conocimiento alcanza, en la realización del trabajo, se han respetado los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

Granada, Junio 2013

El Doctorando:

Fdo.: Javier Sánchez Monedero

Los directores de tesis:

Fdo.: César Hervás Martínez

Fdo.: Pedro Antonio Gutiérrez Peña

MENCIÓN DE DOCTORADO INTERNACIONAL

Esta tesis cumple los criterios para la obtención de la mención «Doctorado Internacional» concedida por la Universidad de Granada. Para ello se presentan los siguientes requisitos:

1. Estancia predoctoral realizada en otros países europeos:
   - **School of Computer Science, The University of Birmingham, Birmingham, Reino Unido.** 3 meses de septiembre a diciembre de 2011.
2. Esta tesis está avalada por los siguientes informes de idoneidad realizados por doctores de otros centros de investigación europeos:
   - **Dr. Peter Tiňo**. Profesor de Escuela de Ciencias de la Computación (*School of Computer Science*) en la Universidad de Birmingham (Reino Unido).
   - **Dr. Huanhuan Chen**. Investigador del Centro de Excelencia para la Investigación en Inteligencia Computacional y aplicaciones (*Centre of Excellence for Research in Computational Intelligence and Applications, (CERCIA)*), centro adjunto a la Universidad de Birmingham (Reino Unido).
   - **Dr. Yoan Miche**. Investigador del Departamento de Información y Ciencias de la Computación (*Department of Information and Computer Science*) de la Universidad de Aalto (Finlandia).
3. La defensa de tesis y el texto se han realizado parcialmente en dos idiomas europeos, español e inglés. La tesis está escrita al completo en inglés y cuenta con un amplio resumen en español.
4. Entre los miembros del tribunal se encuentra un doctor procedente de un centro de educación superior europeo, tratándose del Dr. Xin Yao, Catedrático de la Escuela de Ciencias de la Computación de la Universidad de Birmingham (Reino Unido).

Granada, Junio 2013

El Doctorando:

Fdo.: Javier Sánchez Monedero

Dedicado a mis padres, María Jesús y Luis,
a mi hermana Amaya

al resto de mi familia

y a la familia afectiva

RESUMEN (ABSTRACT IN SPANISH)

El aprendizaje automático (*machine learning*) es una de las ramas de investigación más populares de la inteligencia artificial. El objetivo es desarrollar de manera automática modelos que aprendan de una serie de datos y proporcionen una respuesta sin intervención humana. Las aplicaciones del aprendizaje automático abarcan áreas como robótica, microbiología, biomedicina, agronomía, epidemiología o economía, entre otras muchas. En estos campos, es muy importante la tarea de predecir el valor de una variable de respuesta que puede ser de dos o de múltiples categorías (problemas de clasificación nominal, como, por ejemplo, clasificar terrenos como libres o infestados por malas hierbas para realizar una fumigación selectiva), o también problemas donde la variable toma valores continuos en la recta real (problemas de regresión, como, por ejemplo, la predicción de la velocidad del viento con el fin de diseñar parques eólicos de la mejor forma posible).

Cuando existe una relación de orden entre las categorías de la variable de respuesta, el problema se denomina "clasificación ordinal". La clasificación ordinal (también conocida como regresión ordinal) es un tipo de problema de reconocimiento de patrones que se encuentra situado entre la clasificación nominal y la regresión. De la primera se diferencia en que existe un orden preestablecido entre las clases mientras que de la regresión se distingue en que el conjunto de etiquetas es finito y las diferencias entre los valores de las etiquetas no están definidas.

La clasificación ordinal tiene aplicación en multitud de áreas como la evaluación de la enseñanza [1], evaluación de seguros de coches [2], producción de pasto [3], tratamiento de cáncer de mama [4], predicción de la velocidad del viento [5] o evaluación del crédito [6]. A pesar de sus múltiples aplicaciones, la clasificación ordinal ha recibido poca atención en la comunidad de aprendizaje automático en comparación con los problemas de clasificación nominal. Sin embargo, el número de trabajos relacionados con ésta está aumentando en los últimos años a nivel internacional.

La clasificación ordinal presenta diferentes retos que están abiertos a día de hoy:

REVISIÓN DEL ESTADO DEL ARTE EN REGRESIÓN ORDINAL. En comparación con la clasificación nominal, la regresión ordinal es un campo del aprendizaje automático que ha sido relativamente poco estudiado y explorado. Sin embargo, existen trabajos y publicaciones en la bibliografía que motivan un análisis de los mismos. Especialmente, parece necesario proponer una taxonomía de métodos de regresión ordinal, así como realizar una recopilación de las principales métricas de rendimiento. Estas dos cuestiones ayudarán a contextualizar las propuestas de esta tesis.

DESBALANCEO DE LAS CLASES. Considerando el evidente carácter multiclase y la naturaleza de algunos problemas, las bases de datos ordinales presentan un alto grado de desbalanceo entre las clases (algunas clases tienen muy pocos patrones en comparación con otras), lo que puede provocar que algunos clasificadores ignoren a las clases con un número significativamente menor de patrones, convirtiéndolos en clasificadores triviales para las clases mayoritarias.

EXPLOTACIÓN DE LA RELACIÓN DE ORDEN DE LAS CLASES . Varios autores definen a los clasificadores ordinales y sus algoritmos de entrenamiento como a) métodos que optimizan la clasificación de acuerdo a métricas que consideren el orden y magnitud de los errores y b) métodos que explotan el conocimiento a priori de la disposición ordenada de los patrones en el espacio de entrada. No obstante, el segundo aspecto no suele contemplarse de manera explícita en la formulación de los clasificadores.

Esta tesis trabaja objetivos enunciados en torno a los anteriores retos, aunque se centrará en los dos últimos objetivos, siendo el primero necesario pero no el eje fundamental de la tesis, y el segundo un problema no exclusivo de la clasificación ordinal.

El resultado de este trabajo está reflejado en las publicaciones en conferencias y revistas internacionales asociadas (ver sección específica *Publications*). Además, de manera transversal, la tesis realiza un estudio del estado del arte centrado específicamente en la clasificación ordinal, aunque brevemente se estudian otras cuestiones de interés relacionadas con la clasificación en general.

ABSTRACT

Machine learning is one of the most active branches of artificial intelligence. The purpose is to automate the develop of models that learn from a set of data and provide an output without human interaction. Machine learning application examples are robotics, microbiology, biomedicine, agronomy, epidemiology or finance among others. In all these fields it is important to predict the value of a response variable that can have two or multiple scales (for instance, nominal classification problems, such as the classification of areas as weed-free or infested in order to perform a selective treatment of weeds) or problems where the target variable is continuous (regression problems, such as wind speed forecasting for wind farms set up).

When there exists an order relationship in the class variable, the problem is named to as "ordinal regression" (also known as ordinal classification). The samples are labelled into a set of category labels with an ordering amongst the categories. In contrast to nominal classification, there is an ordinal relationship throughout the categories and it is different from regression in that the number of ranks is finite and exact amounts of difference between ranks are not defined. In this way, ordinal classification lies somewhere between nominal classification and regression.

Ordinal classification problems are important, since they are common in our everyday life where many problems require classification of items into naturally ordered classes. Examples of these problems are the teaching assistant evaluation [1], car insurance risk rating [2], pasture production [3], preference learning [7], breast cancer conservative treatment [4], wind forecasting [5] or credit rating [6]. However, compared with general classification problems, much less effort has been devoted to ordinal classification learning. Nevertheless, in the last decade an increasing number of publications report progress in the artificial learning of ordinal concepts.

Nowadays, ordinal regression presents some challenges that form the research line of the present thesis:

STATE-OF-THE-ART IN ORDINAL REGRESSION. Compared with nominal classification, ordinal regression is a machine learning field much less studied and explored. However, there are several works and literature dealing with this issue, which makes necessary to perform a proper analysis of them. A taxonomy of ordinal regression methods and an effort to gather and compare the main metrics for their evaluation would help to contextualize the proposals in the field.

CLASS IMBALANCE. Considering the obvious multi-class feature and some of the ordinal classification problems nature, the ordinal regression datasets present a high imbalance degree (i.e. some classes have few patterns when compared to the rest of the classes). Imbalance problem can generally harm classifiers, which tend to ignore minority populated classes, presenting a trivial behaviour with respect to those classes.

DATA ORDERING EXPLOITATION. Several authors define ordinal classifiers, and their associated learning algorithms, as a) methods that optimize the classification

task constrained to metrics that consider errors magnitude, and b) methods which exploit the a priori knowledge about ordered placement of patterns according to their class in the input space. However, the second aspect of ordinal classifiers is not usually included explicitly in the classifier formulation.

The present thesis deals with objectives related to the aforementioned challenges, though the two second challenges are the most relevant ones for the thesis.

The work done in this thesis is reflected in several international conferences and journals (see Publications section).

PUBLICATIONS

The ideas of this thesis have appeared previously in the following publications (other publications done during the PhD. can be seen at http://www.uco.es/grupos/ayrna/jsanchezm):

- INTERNATIONAL JOURNAL PAPERS
  - J. Sánchez-Monedero, C. Hervás-Martínez, P.A. Gutiérrez, M. Carbonero-Ruz, M. C. Ramírez-Moreno, and M. Cruz-Ramírez. Evaluating the Performance of Evolutionary Extreme Learning Machines by a Combination of Sensitivity and Accuracy Measures. *Neural Network World*, 20:899–912, 2010.
    Impact factor (JCR2010): 0.511
    http://www.nnw.cz/obsahy10.html
  - J. Sánchez-Monedero, P. A. Gutiérrez, F. Fernández-Navarro and C. Hervás-Martínez. Weighting efficient accuracy and minimum sensitivity for evolving multi-class classifiers. *Neural Processing Letters*, 34 (2):101–116, 2011.
    Impact factor (JCR2011): 0.750
    http://dx.doi.org/10.1007/s11063-011-9186-9
  - J. Sánchez-Monedero, P.A. Gutiérrez, P. Tiňo and C. Hervás-Martínez. Exploitation of pairwise class distances for ordinal classification. *Neural Computation*, Accepted:MIT Press, 2013.
    JCR: 1.884
    http://dx.doi.org/10.1162/NECO_a_00478
  - P.A. Gutiérrez, S. Salcedo-Sanz, C. Hervás-Martínez, L. Carro-Calvo, J. Sánchez-Monedero and L. Prieto. Ordinal and nominal classification of wind speed from synoptic pressure patterns. *Engineering Applications of Artificial Intelligence*, 26(3):1008–1015, 2013.
    Impact factor (JCR2011): 1.665
    http://dx.doi.org/10.1016/j.engappai.2012.10.018
  - M. Cruz-Ramírez, C. Hervás-Martínez, J. Sánchez-Monedero and P.A. Gutiérrez. Metrics to guide a multi-objective evolutionary algorithm for ordinal classification. *Neurocomputing*. 2013 (Accepted)
    Impact factor (JCR2011): 1.580

  Some ideas have been submitted to different journals and are under review process:

  - J. Sánchez-Monedero, P. Campoy-Muñoz, P.A. Gutiérrez and C. Hervás-Martínez. A guided data projection technique for classification of sovereign ratings: the case of European Union 27. *Applied Soft Computing* (Under Review).
  - P.A. Gutiérrez, M. Pérez-Ortiz, J. Sánchez-Monedero, F. Fernández-Navarro and C. Hervás-Martínez. Ordinal regression methods: survey and experi-

mental study. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (Under Review).

- INTERNATIONAL CONFERENCE PUBLICATIONS:

  - J. Sánchez-Monedero, C. Hervás-Martínez, F. Martínez-Estudillo, Mariano Carbonero, M. Moreno, and M. Cruz-Ramírez. Evolutionary learning using a sensitivity-accuracy approach for classification. In *Hybrid Artificial Intelligence Systems*, volume 6077 of *Lecture Notes in Computer Science*, pages 288–295. Springer Berlin / Heidelberg, 2010.
    http://dx.doi.org/10.1007/978-3-642-13803-4_36
  - J. Sánchez-Monedero, M. Cruz-Ramírez, F. Fernández-Navarro, J.C. Fernández, P.A. Gutiérrez, and C. Hervás-Martínez. On the suitability of extreme learning machine for gene classification using feature selection. In *Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on*, pages 507–512, 2010.
    http://dx.doi.org/10.1109/ISDA.2010.5687215
  - J. Sánchez-Monedero, M. Carbonero-Ruz, D. Becerra-Alonso, F. Martínez-Estudillo, P.A. Gutiérrez, and C. Hervás-Martínez. Numerical variable reconstruction from ordinal categories based on probability distributions. In *Proceedings of the 11th International Conference on Intelligent Systems Design andApplications (ISDA 2011)*, pages 1182–1187, Cordoba, Spain, Spain, nov 2011.
    http://dx.doi.org/10.1109/ISDA.2011.6121819
  - M. Cruz-Ramírez, César Hervás-Martínez, J. Sánchez-Monedero, and P.A. Gutiérrez. A Preliminary Study of Ordinal Metrics to Guide a Multi-Objective Evolutionary Algorithm. In *Proceedings of the 11th International Conference on Intelligent Systems Design and Applications (ISDA 2011)*, pages 1176–1181, Cordoba, Spain, 2011.
    http://dx.doi.org/10.1109/ISDA.2011.6121818
  - P.A. Gutiérrez, Sancho Salcedo-Sanz, César Hervás-Martínez, Leo Carro-Calvo, J. Sánchez-Monedero, and Luis Prieto. Evaluating nominal and ordinal classifiers for wind speed prediction from synoptic pressure patterns. In *Proceedings of the 11th International Conference on Intelligent Systems Design andApplications (ISDA 2011)*, pages 1265–1270, Cordoba, Spain, Spain, nov 2011.
    http://dx.doi.org/10.1109/ISDA.2011.6121833
  - P.A. Gutiérrez, M. Pérez-Ortiz, F. Fernández-Navarro, J. Sánchez-Monedero and C. Hervás-Martínez. An Experimental Study of Different Ordinal Regression Methods and Measures. In *7th International Conference on Hybrid Artificial Intelligence Systems*, pages 296–307, 2012.
    http://dx.doi.org/10.1007/978-3-642-28931-6_29
  - J. Sánchez-Monedero, P.A. Gutiérrez, M. Pérez-Ortiz and C. Hervás-Martínez. An n-spheres based synthetic data generator for supervised classification. In *International Work-Conference on Artificial Neural Networks (IWANN)*, volume 7902 of *Lecture Notes in Computer Science*, pages 613–621. Springer-Verlag Berlin Heidelberg, 2013.

- J. Sánchez-Monedero, P.A. Gutiérrez and C. Hervás-Martínez. Evolutionary ordinal extreme learning machine. *International Conference on Hybrid Artificial Intelligence Systems (HAIS 2013)* (Accepted).

- SOFTWARE RELEASED AS FREE SOFTWARE:

  - Source code of Pairwise Class Distance Ordinal Classifier (PCDOC) algorithm (including real and synthetic datasets, and code examples) is released under GPL licence at:
    http://www.uco.es/grupos/ayrna/neco-pairwisedistances
  - Source code of the synthetic data generator based on n-spheres is released under GPL licence at:
    http://www.uco.es/grupos/ayrna/iwann2013-syntheticdatagenerator

FUNDING

Funding for my Ph.D. studies has been supplied by "Junta de Andalucía" Ph. D. program with is partially funded by the European Social Fund.

This work has been partially subsidized by the TIN2008-06681-C06-03 and TIN2011-22794 projects of the Spanish Ministerial Commission of Science and Technology (CICYT), FEDER funds and the P08-TIC-3745 and P11-TIC-7508 projects of the "Junta de Andalucía" (Spain).

FINANCIACIÓN

Los estudios de doctorado del doctorando han sido financiados por el programa de formación de investigadores de la Junta de Andalucía, que está parcialmente financiado por el Fondo Social Europeo.

Este trabajo ha sido subvencionado parcialmente por los proyectos TIN2008-06681-C06-03 y TIN2011-22794 de la Comisión Interministerial de Ciencia y Tecnología (CICYT) y con fondos FEDER. También ha sido subvencionado parcialmente por los Proyectos de Excelencia P08-TIC-3745 y P2011-TIC-7508 de la Junta de Andalucía.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

computational intelligence	Is a branch of computer science studying problems for which there are no effective computational algorithms. 25
confusion matrix	Also known as *contingency table* or *error matrix*, is a square matrix allowing visualization of classifier performance. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. 81, 83, 85, 92, 159
differential evolution	Differential evolution (DE) is a type of evolutionary algorithms that optimizes a problem by managing a population of candidate solutions and create new candidate solutions by combining them with the existing ones, the best candidates are typically kept during the evolution process. 80, 88, 90, 97–99
experiments	Process of model validation and comparison that involves among others data partition, validation procedures and statistical comparison test. 94, 97, 98, 110, 115, 151, 155, 164
Extreme Learning Machine	Extreme Learning Machine (ELM) is a type of neural networks where the learning process is done without iterative tuning. 33, 67, 80, 89, 91
imbalanced dataset	A dataset where at least one of the classes (often called *minority class*) contains a much smaller number of patterns than the other classes (often referred as *majority classes*). Also named to as a dataset with *class imbalance*. 29, 60, 81, 88, 139, 165, 175
latent variable	latent variables (as opposed to observable variables) are variables that are not directly observed but are rather inferred (through a mathematical model) from other variables that are observed (directly measured). 30, 68, 106, 120, 176

machine learning — Field of study that gives computers the ability to learn without being explicitly programmed. 25, 81, 99, 126, 148, 149, 166, 167

Minimum Sensitivity — The minimum sensitivity (MS) of the classifier is defined as the minimum value of the sensitivities for each class, being the sensitivity of a class the rate of patterns correctly predicted for a class with respect to the total number of patterns in that class. 80, 85, 86, 94, 97

neural network — An *artificial neural network* (ANN), often just named a *neural network*, is a mathematical model inspired by biological neural networks. A neural network consists of an interconnected group of artificial neurons, and it processes information using a connectionist approach to computation. 25, 33, 37, 38, 67, 87, 88, 97–99, 110, 150, 162, 176

ordinal regression — Ordinal regression (OR) is a supervised learning problem of predicting categories that have an ordered arrangement (also named as ordinal classification). In this thesis ordinal regression is shorted as OR. 28, 55, 57, 58, 62, 106, 148

Support Vector Machines — Support Vector Machines (SVM) are supervised learning models that aims to find the maximum-margin separating hyperplane that allows to classify two classes. They are extended tor regression and multi-class cases.. 25, 37, 44, 49, 56, 63, 70, 76, 87, 150, 162, 167, 169

threshold methods — Also known as variable methods these methods assume that ordinal response is a coarsely measured latent continuous variable and model it as real intervals in one dimension. 30, 68, 106, 113, 119

ACRONYMS

Acc	Accuracy. 28, 60, 80, 81, 83, 85, 86, 155
AMAE	Averaged Mean Absolute Error. 28, 57, 60, 110, 155
ASAOR	A Simple Approach to Ordinal Regression. 76, 132, 154, 167
AUC	area under the ROC curve. 83, 84, 150
CCR	Correct Classification Rate. 28, 44, 60, 168
E-ELM	Evolutionary Extreme Learning Machine. 16, 80, 90, 92, 110, 176
E-ELM-CS	Evolutionary ELM considering C and MS. 80, 92, 94
E-ELMOR	Evolutionary Extreme Learning Machine for Ordinal Regression. 16, 106, 108, 110, 176, 178
ELMOR	Extreme Learning Machine for Ordinal Regression. 20, 107, 108, 110, 180
GPOR	Gaussian Processes for Ordinal Regression. 72, 76, 117, 132, 154, 167, 170
KDLOR	Kernel Discriminant Learning for Ordinal Regression. 20, 72, 76, 127, 132, 180
MAE	Mean Absolute Error. 28, 60, 110, 118, 155, 168
MLP	MultiLayer Perceptron. 40, 41, 87, 89, 97, 98, 155
MOEA	multi-objective evolutionary algorithm. 88, 97, 99
MSE	Mean Squared Error. 85, 88, 97
MZE	Mean Zero-one Error. 60, 110, 118
NVR	Numerical Variable Reconstruction. 17, 106, 113, 115, 177
OR	ordinal regression. 15, 28, 106, 175, 180
PCD	Pairwise Class Distance. 18, 105, 106, 119–121, 177, 179

PCDOC	Pairwise Class Distances for Ordinal Classification. 18, 20, 106, 119, 120, 145, 147, 148, 154, 177, 180
POM	Proportional Odds Model. 69, 76, 132
RED-SVM	Reduction from cost-sensitive ordinal ranking to weighted binary classification (RED) with SVM. 73, 76, 118, 132, 154, 167, 169
RMSE	Root Mean Squared Error. 112
ROC	Receiver Operating Characteristic. 83, 84, 176
S	Sensitivity. 82, 85
SVC	Cost Support Vector Classification. 20, 118, 127, 132, 180
SVOREX	Support Vector Ordinal Regression with explicit constraints. 71, 76, 127, 154, 167, 169
SVORIM	Support Vector Ordinal Regression with implicit constraints. 20, 71, 76, 118, 132, 154, 167, 169, 180
SVR	Support Vector Regression. 51, 118

RESUMEN EN CASTELLANO (SUMMARY IN SPANISH)

Esta primera parte de la tesis corresponde a la traducción al castellano de los capítulos de introducción y conclusiones de la misma, tal y como recoge la normativa para la obtención de la Mención de Doctor Internacional, desarrollada en el artículo 19 de las Normas reguladoras de las enseñanzas oficiales de Doctorado y del título de Doctor por la Universidad de Granada aprobadas por Consejo de Gobierno de la Universidad de Granada en su sesión del 2 de Mayo del 2012. El capítulo de introducción presenta el problema de la regresión ordinal dentro del campo del aprendizaje automático. El capítulo de conclusiones realiza un breve resumen de las aportaciones de la tesis y propone algunas líneas de trabajo futuro.

This first part of the thesis presents a Spanish summary of the dissertation.

0

RESUMEN DE LA TESIS

Resumen. Este capítulo de resumen en castellano presenta la traducción del inglés de los capítulos de introducción y conclusiones de la tesis. Inicialmente se introduce el *aprendizaje automático* (o *machine learning*) como base para poder presentar el problema de la regresión ordinal (RO). La regresión ordinal corresponde a un tipo de problemas que pertenecen a las técnicas de clasificación supervisada, también conocida como «algoritmos de aprendizaje» o incluso también como «clasificadores». En este capítulo se hace una introducción breve al problema del aprendizaje supervisado y a la regresión ordinal para contextualizar y motivar al lector antes de establecer los principales objetivos de la tesis.

En las últimas secciones se realiza una discusión y elaboración de conclusiones. A grandes rasgos, en esta tesis se abordan los siguientes grandes objetivos: realizar un estudio de revisión del estado del arte para RO, proponer y desarrollar nuevo métodos de RO en torno a temas relacionados con la RO y aplicar las técnicas de RO a problemas reales. En nuestra opinión, estos objetivos se ha alcanzado, tal y como sintetiza este capítulo de resumen.

0.1 APRENDIZAJE AUTOMÁTICO Y APRENDIZAJE SUPERVISADO

El *aprendizaje automático* (o *machine learning* en inglés) no tiene una definición clara, tal y como señala el Profesor Andrew Ng [10] en uno de sus cursos de introducción a esta materia. En 1959, Arthur Samuel definió el aprendizaje automático como el «campo de estudio que proporciona a los ordenadores la capacidad de aprender sin haber sido explícitamente programados» [10, 11]. Más tarde, Tom M. Mitchell propuso que «un programa informático se dice que aprende de una experiencia E con respecto a una clase de tareas T y una medida de rendimiento P, si su rendimiento en las tareas del tipo T, medida por P, mejora con la experiencia E» [12]. Así, podemos afirmar que el aprendizaje automático equivale a «aprender de los datos» con el fin de extraer el conocimiento necesario según diferentes propósitos, como por ejemplo ayudar las personas en procesos de decisión o incluso automatizar totalmente decisiones a partir de los datos, así como adaptar sistemas de manera dinámica para mejorar las experiencias del usuario [13]. Este «aprender de los datos» hace que el aprendizaje automático se sitúe entre diferentes ramas que pertenecen a la inteligencia artificial, la estadística y las matemáticas (ver Figura 0.1). Dependiendo de cómo se realice este aprendizaje se han desarrollado diferentes disciplinas, aunque probablemente la más activa sea la conocida como *inteligencia computacional* (o *computational intelli-*

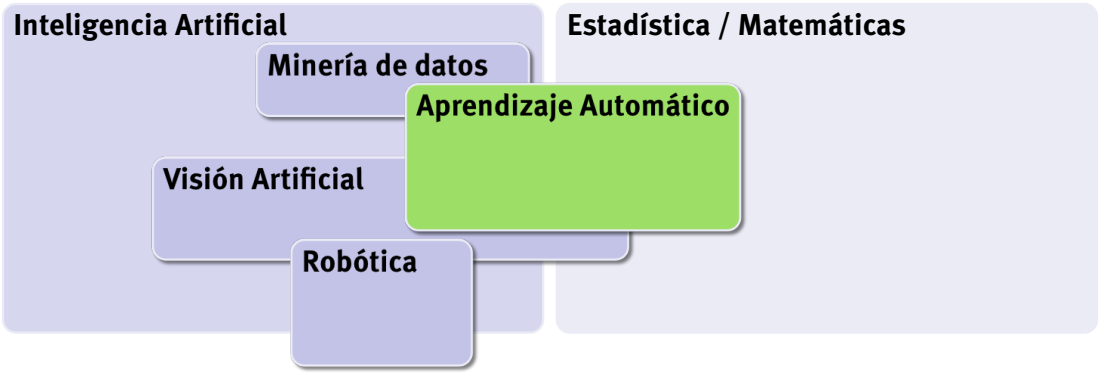


Figura 0.1: Aprendizaje automático: dónde encaja y dónde no (fuente [13]).

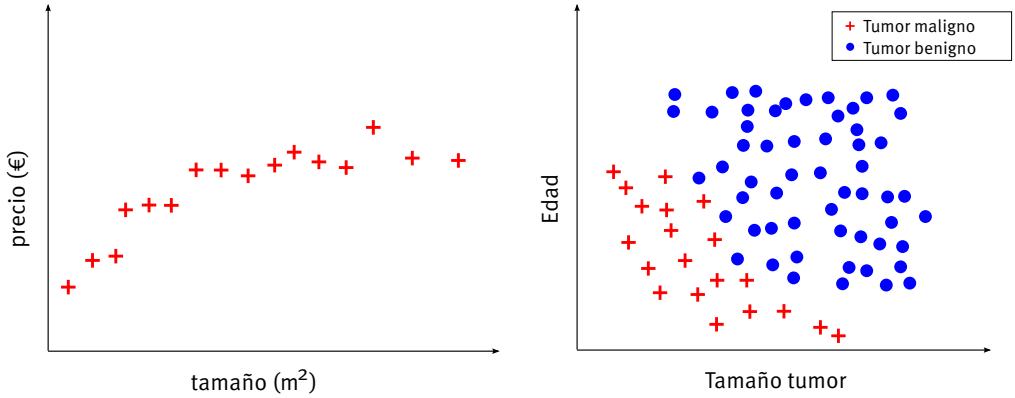
gence) [14]. La inteligencia computacional engloba a las redes neuronales artificiales [15, 16], los sistemas difusos y la computación evolutiva, incluida la inteligencia de enjambres. Por motivos históricos, algunos métodos estadísticos y matemáticos más recientes, como las máquinas de vectores soporte (*support vector machines*, SVM), las redes Bayesianas y el razonamiento probabilístico, o incluso el procesamiento estadístico del lenguaje natural, han sido incluidos en conferencias y revistas enfocadas a la inteligencia computacional, mientras que otros como las técnicas de optimización numérica, la teoría de aproximación, métodos estadísticos o la lógica de primer orden están fuera de su ámbito [14].

Uno de los campos del aprendizaje automático más activos es el de *reconocimiento de patrones* (*pattern recognition*), que a grandes rasgos se puede dividir entre *aprendizaje supervisado* o *aprendizaje no supervisado*. El aprendizaje supervisado puede describirse informalmente como enseñar al ordenador a hacer algo para que después el ordenador pueda seguir haciéndolo a partir del conocimiento descubierto durante el aprendizaje. Por otro lado, en el aprendizaje no supervisado el ordenador aprende a hacer algo, y utilizamos este conocimiento para tratar de determinar tanto la estructura como los patrones de los datos. En el primer caso se proporciona al ordenador un conjunto de datos *etiquetados* mientras que en el segundo caso se proporcionan datos *no etiquetados* que el ordenador tiene que estructurar.

El aprendizaje supervisado tal vez sea el tipo de problema de aprendizaje automático más común. Por ejemplo, supongamos que queremos predecir el precio de una vivienda a partir de su tamaño en metros cuadrados tal y como se muestra en la Figura 0.2a. En este caso podemos recoger datos de precios de viviendas y su tamaño en metros cuadrados, obteniendo así lo que se denomina como *conjunto de entrenamiento*, que está compuesto por las variables independientes del sistema (el tamaño en metros cuadrados, m^2), y las variables dependientes[1] o «respuestas correctas» (etiquetas). El algoritmo de aprendizaje tiene que construir un modelo a partir de estos datos etiquetados disponibles en el conjunto de entrenamiento con el objetivo de poder predecir el precio correcto, que desconocemos, para nuevos datos no vistos durante el aprendizaje. Este ejemplo en el que la variable que queremos predecir es de naturaleza continua y perteneciente a la recta real es un tipo de problema conocido como *regresión*. Los datos no vistos durante el proceso de aprendizaje, y que

1 Normalmente sólo habrá una variable dependiente, aunque existen ramas de la clasificación supervisada como la clasificación multi-etiqueta en las que existe más de una variable dependiente.

se utilizan para comprobar el rendimiento real del modelo de predicción, se suelen denominar como *datos de generalización* o *datos de test*.



(a) Ejemplo de problema de regresión:«Dados estos datos, un amigo tiene una casa de 75 metros cuadrados, ¿por cuánto podría esperar venderla?».

(b) Ejemplo de problema de clasificación «¿Podrías estimar un diagnóstico basado en el tamaño del tumor y la edad del paciente?»

Figura 0.2: Ejemplos de problemas de regresión y clasificación (fuente [10]).

Por otro lado, cuando la variable para predecir es discreta, el problema se llama *problema de clasificación*. Por ejemplo, la Figura 0.2b muestra un problema de clasificación en el que queremos discriminar si un cáncer de pecho es maligno o benigno basándonos en la edad del paciente y en el tamaño del tumor[2]. A partir de los datos de entrenamiento el algoritmo de aprendizaje debería construir un modelo capaz de distinguir las dos clases («maligno» o «benigno»). La variable de salida puede tener más de dos valores, por ejemplo tumor «benigno», tumor «maligno tipo A», tumor «maligno tipo B» y tumor «maligno tipo C». En el primer caso se denomina problemas binarios y en el segundo problemas multi-clase. Existe un tercer tipo de problemas de clasificación que de hecho ocupan el foco de interés de esta tesis: si existe una relación de orden entre las clases, los problemas se denominan problemas de *clasificación ordinal*. Por ejemplo, si queremos clasificar los tipos de un tumor como «benigno», «sospechoso maligno», «maligno» o «maligno grave» probablemente estemos hablando de un problema de clasificación ordinal ya que estamos tratando con diferentes grados de una enfermedad. La *clasificación ordinal* también se conoce como *regresión ordinal* ya que tiene tanto relación con la clasificación como con la regresión, tal y como se explicará en el siguiente apartado.

La Figura 0.3 nos muestra un ejemplo de un problema de clasificación binario comparado con un problema de clasificación ordinal donde el objetivo es detectar una enfermedad. En el primer caso, el clasificador sólo será capaz de detectar la presencia o ausencia de una enfermedad. En el segundo caso, el clasificador podrá detectar diferentes grados de la enfermedad. Llegados a este punto podríamos hablar de clasificación nominal multi-clase, sin embargo hay varios puntos que diferencian a la clasificación ordinal de la nominal: dada la naturaleza de problema, existe una disposición de orden entre las etiquetas (clases), y este orden puede estar también

2 Estas variables se han elegido a modo de ejemplo, obviamente no es posible clasificar el tipo de tumor únicamente a partir de estas dos variables.

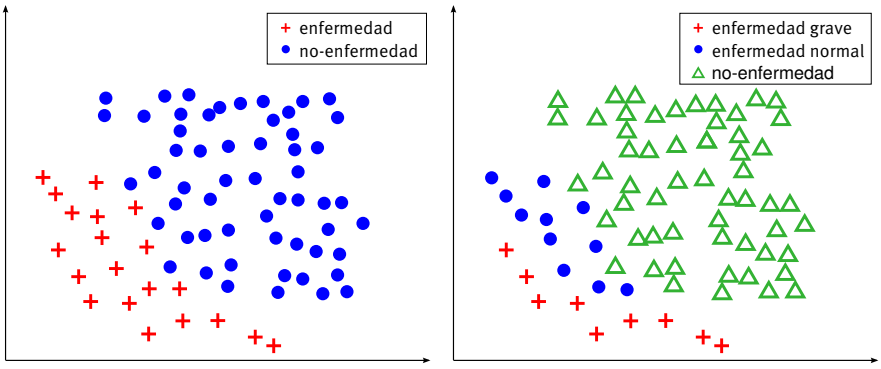


Figura 0.3: Un ejemplo de clasificación binaria (figura a la izquierda) frente a la clasificación ordinal (figura de la derecha). En el primer caso hay dos estados para un patrón: enfermo o no enfermo. Sin embargo, un experto que se apoye en técnicas de aprendizaje automático puede demandar grados de clasificación más finos, en cuyo caso podría afrontarse el problema como clasificación multi-clase. Si las clases tienen una relación de orden entre ellas estamos tratando con un problema de regresión ordinal, que es un enfoque que aporta mayor precisión en este ejemplo.

presente en el espacio de atributos o variables de entrada[3]. Esto afecta al clasificador de dos formas: primero, el clasificador debe explotar este conocimiento a priori sobre la distribución de patrones en el espacio de entrada [17], y, segundo, la evaluación del clasificador necesita medidas de rendimiento específicas [18]. Esto último se traduce en que un error de clasificación de un patrón de la clase «enfermedad grave» clasificado por el modelo como «sano» debe ser más penalizado que si el modelo clasifica ese mismo patrón como «enfermedad normal». Estos dos retos marcan las diferencias principales con la clasificación nominal, e inducen a pensar que afrontar problemas como los del ejemplo anterior desde el punto de vista de la regresión ordinal puede tanto ayudar a mejorar los resultados como contribuir a la motivación última del aprendizaje automático de «*aprender de los datos*».

0.2 CLASIFICACIÓN ORDINAL

La *clasificación ordinal*, o *regresión ordinal*, es un problema de clasificación supervisada en el que el objetivo es predecir la categoría a la que pertenece un patrón habiendo una relación de orden entre las categorías. Además, cuando el problema manifiesta claramente una naturaleza ordinal, se espera que este orden esté presente de alguna manera en el espacio de entrada de los datos [17]. Los datos se etiquetan de acuerdo a un conjunto de niveles de forma que se establece un orden entre los mismos. La regresión ordinal se diferencia de la clasificación nominal en que existe una relación de orden entre las categorías; y se diferencia de la regresión estándar en que el número de niveles es finito, y la diferencia entre estos niveles no está definida. De esta forma, la clasificación ordinal se sitúa entre la clasificación y la regresión.

Este tipo de problemas de clasificación no debe confundirse con problemas de ordenación (*sorting*) o de clasificación con rango (*ranking*). Los problemas de ordenación pretenden relacionar todos los patrones del conjunto de generalización respecto a un

3 Aunque puede suceder que la naturaleza de un problema sugiera una relación de orden entre las etiquetas que no se traslada al espacio de entrada.

orden total. La clasificación con rango se refiere a ordenar los patrones con un orden relativo. La clasificación ordinal también puede utilizarse para ordenar patrones, pero el objetivo es obtener una buena precisión de clasificación a la vez que se mantiene el orden de los patrones.

La importancia de la clasificación ordinal es obvia si pensamos que el tipo de problemas donde es necesario clasificar patrones en clases ordenadas son habituales. Ejemplos de estos problemas son la evaluación del apoyo al aprendizaje [1], evaluación de seguros de vehículos [2], producción de pasto [3], problemas del área de aprendizaje con preferencias [7], tratamiento conservativo de cáncer de pecho [4] o la evaluación de créditos [6].

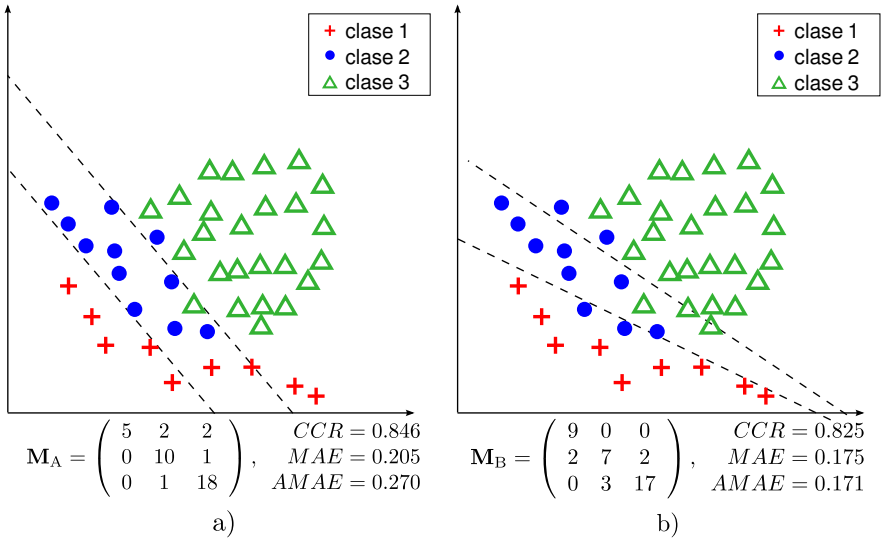


Figura 0.4: Ejemplo de de la necesidad de usar métricas de evaluación alternativas para los clasificadores ordinales. Supongamos que tenemos un problema de clasificación de una enfermedad donde la clase 1 significa «enfermedad grave», la clase 2 significa «enfermedad normal» y la clase 3 significa «ausencia». En el ejemplo, el mismo conjunto de puntos es clasificado por los clasificadores a) y b). $\mathbf{M}_A$ y $\mathbf{M}_B$ son las correspondientes matrices de confusión. El clasificador a) tiene el mejor rendimiento para la métrica CCR, sin embargo, el clasificador b) tiene mejor rendimiento si se consideran las métricas MAE y AMAE. En este caso, aunque a) obtiene el mejor ratio de clasificación, está cometiendo graves errores porque está clasificando patrones de la clase 1 («enfermedad grave») con patrones de la clase 3 («ausencia»). Las métricas MAE y AMAE mejoran para el clasificador b) señalando que el clasificador b) es un clasificador ordinal mejor que a) para esta muestra de patrones en concreto.

Los problemas de clasificación ordinal presentan dos grandes cuestiones que deben considerarse para el diseño de los algoritmos de aprendizaje. En primer lugar, la naturaleza del problema indica que el orden de las clases debe estar relacionado de alguna forma con la distribución de patrones en el espacio de atributos, así como en la distribución topológica de las clases (aunque, en general, esta relación será una relación no lineal). En consecuencia, un clasificador debe explotar este conocimiento a priori sobre el espacio de entrada [17]. En segundo lugar, cuando se evalúa el rendimiento de un clasificador ordinal, las métricas de rendimiento empleadas deben considerar el orden de las clases, de tal forma que los errores de clasificación entre clases

adyacentes deben ser considerados como menos importantes que errores de clasificación entre clases no adyacentes (más separadas en la escala ordinal). Por ejemplo, consideremos un conjunto de datos de predicción del tiempo con la variable objetivo tomando valores en el conjunto $\{$*muy frío*, *frío*, *templado*, *caluroso*, *muy caluroso*$\}$, con una clara relación de orden natural entre las clases, *muy frío* $\prec$ *frío* $\prec$ *templado* $\prec$ *caluroso* $\prec$ *muy caluroso*. Es evidente que predecir erróneamente la clase *caluroso* cuando la clase real es *frío* representa un error más grave que el error asociado a predecir *muy frío* [18]. Así, son necesarias medidas de rendimiento específicas para evaluar el rendimiento de un clasificador ordinal [18–20].

Mientras que la *precisión*, también conocida como ratio de clasificación correcta (*Correct Classificacion Ratio*, CCR), es la métrica más común para la clasificación nominal, el error absoluto medio (*Mean Absolute Error*, MAE) es la medida más común en el contexto de regresión ordinal. Además, en clasificación multi-clase se han propuesto diversas medidas alternativas, por ejemplo, para evaluar el rendimiento individual en las clases [21, 22] (incluyendo aquellas clases peor clasificadas). Análogamente, el MAE medio (*average MAE*, AMAE) ha sido propuesto por Baccianella et al. [23] para evaluar de forma más precisa el rendimiento en bases de datos desbalanceadas (aquellos problemas de clasificación donde el número de patrones de cada clase es muy diferente). La Figura 0.4 presenta un ejemplo motivador de un problema de clasificación ordinal done queremos comparar el rendimiento de dos clasificadores sobre el mismo conjunto de datos. En el ejemplo, el clasificador b) es mejor clasificador ordinal que el clasificador a), a pesar de que la precisión global de a) es mayor.

0.3 BREVE ESTADO DEL ARTE EN REGRESIÓN ORDINAL

En los últimos años han surgido bastantes propuestas relacionadas con la clasificación ordinal. Por ejemplo, Raykar et al. [24] han diseñado funciones de clasificación por rangos en el contexto de la regresión ordinal y el filtrado de información colaborativo. Kramer et al. [25] han mapeado la escala ordinal asignando variables numéricas para luego aplicar un modelo de árbol de regresión.

Otras alternativas simples que aparecen en la literatura tratan de imponer la estructura ordinal a través de la clasificación sensible a costes, donde clasificadores estándares (nominales) tienen en cuenta la información ordinal mediante la distinta penalización a los errores, normalmente mediante un coste igual a la desviación absoluta entre la etiqueta predicha y la real [26].

La tercera alternativa directa sugerida en la literatura consiste en transformar el problema de clasificación ordinal en un problema de clasificación binaria anidada [27, 28] para combinar, en una segunda fase, las predicciones de los clasificadores en la predicción final ordinal. Es obvio que la información ordinal permite comparaciones entre las diferentes etiquetas de clase. Para una etiqueta concreta k, una pregunta asociada podría ser «¿es la etiqueta del patrón $\mathbf{x}$ mayor que k?». Esta pregunta es equivalente a un problema de clasificación binaria, de modo que un problema de clasificación ordinal puede resolverse aproximando cada problema de clasificación binaria de manera independiente y combinando las salidas binarias para predecir una clase [27].

Otra alternativa [28] impone pesos de forma explícita sobre los patrones de cada sistema binario de forma que los errores en los patrones de entrenamiento se penali-

zan de forma proporcional a la diferencia absoluta entre su rango y k. La binarización de los problemas de regresión ordinal puede enfocarse también como problemas de clasificación binaria aumentada, esto es, los problemas binarios no se resuelven de forma independiente, y en cambio un único clasificador binario se construye para todos los subproblemas. Por ejemplo, Cardoso y Pinto da Costa [29] añaden dimensiones adicionales y replican los puntos de los datos mediante lo que denominan *método de replicación de datos*. Este espacio aumentado se utiliza para construir un clasificador binario, y la proyección en el espacio original resulta en un clasificador ordinal. Un marco de trabajo muy interesante en este sentido es el propuesto por Li y Lin [30], Lin y Li [31], que realiza una reducción desde una clasificación ordinal con costes a un problema de clasificación binaria ponderada (*reduction from cost-sensitive ordinal ranking to weighted binary classification*, RED), que es capaz de reformular el problema como un problema binario utilizando una matriz para extender los patrones originales, un esquema de ponderación y una matriz de costes en forma de V. Una característica interesante de esta propuesta es que unifica muchos de los algoritmos de *ranking* o de regresión ordinal, como el Perceptrón *ranking* [32] o las máquinas de vectores soporte para regresión ordinal [33]. Más recientemente el aprendizaje por cuantificación vectorial (*learning vector quantization*, LVQ) ha sido adaptado al caso ordinal en el contexto de aprendizaje basado en prototipos [34]. En este trabajo la información de orden se utiliza para seleccionar y adaptar prototipos para cada clase, así como para mejorar el proceso de modificación de los prototipos.

Otras propuestas de métodos para clasificación ordinal – de hecho la mayoría de ellas – pueden agruparse bajo el grupo de los *modelos de umbral* [35]. Estos métodos suponen que la respuesta ordinal está asociada a una variable latente medida en escala continua y modelada mediante intervalos de clase sobre la recta real. Basándose en esta asunción, los algoritmos buscan una dirección sobre la que los patrones son proyectados, y fijan una serie de umbrales que parten la dirección de la proyección en intervalos consecutivos que representan las categorías ordinales [35–39].

0.4 MOTIVACIÓN Y RETOS

Tal y como se ha mencionado, la clasificación ordinal trata con problemas de clasificación supervisada en los que existe un orden entre las categorías. Este orden se suele deducir por la naturaleza del problema por parte de un experto o por simples deducciones de los datos. Por ejemplo, un problema de detección de una enfermedad puede afrontarse como un problema de clasificación ordinal en el que el objetivo es asignar la etiqueta de clase adecuada a un paciente a partir de unas variables de entrada, siendo el conjunto de categorías $\{C_1 = \textit{riesgo}, C_2 = \textit{severo}, C_3 = \textit{normal}, C_4 = \textit{posible presencia}, C_5 = \textit{ausencia}\}$, donde las etiquetas representan el grado de una enfermedad asignado por médicos. Así, tenemos una relación de orden natural entre las clases ($\textit{riesgo} \prec \textit{severo} \prec \textit{normal} \prec \textit{posible presencia} \prec \textit{ausencia}$), done *riesgo* es el peor grado de la enfermedad y *ausencia* el mejor. En este caso, la naturaleza ordinal del problema puede deducirse no sólo por la observación de las variables dependientes, las etiquetas de clase, sino también por la observación de las variables independientes, que reflejan este orden en el espacio de atributos $\mathcal{X}$. Es relativamente obvio que variables como la fiebre o el pulso son variables que pueden crecer o decrecer en relación a la etiqueta de clase.

Hasta ahora se han mencionado dos temas de interés en clasificación ordinal: la caracterización de errores y la explotación de la información de orden. Además, y partiendo del ejemplo de detección de enfermedad, existen otras características de los datos que son reconocidas ampliamente como retos dentro de la comunicad de aprendizaje automático. En primer lugar, el problema del desbalanceo de clases también está presente en los problemas de clasificación ordinal [17, 19, 40]. En el ejemplo anterior podríamos afirmar que el número de patrones de la clase *riesgo* puede ser relativamente menor comparado con el número de patrones de la clase *ausencia*. En segundo lugar, puesto que a menudo las etiquetas de clase representan grados o rangos, es bastante probable que se de el caso de solapamiento entre clases o entre subconjuntos de las clases. De nuevo partiendo del ejemplo anterior, es probable que las clases *riesgo* y *severo* estén altamente solapadas. Especialmente para los modelos de umbral, que son los más extendidos [35, 40], la presencia de alta dimensionalidad junto a la separabilidad no lineal de los datos pueden hacer que la función de proyección ϕ resulte en modelos complejos que impongan transformaciones altamente no lineales desde el espacio de entrada al espacio latente. Esta imposición de modelos demasiado rígidos para las proyecciones puede derivar en problemas para clasificar los patrones en las fronteras de la clase, sobre todo en presencia de ruido o solapamiento entre clases, ya que los patrones situados en estas fronteras pueden ser erróneamente proyectados a un intervalo correspondiente a una clase que no es la suya.

Considerando las cuestiones planteadas hasta ahora, podemos sintetizar los siguientes retos abiertos que forman parte de los objetivos de la tesis:

REVISIÓN DEL ESTADO DEL ARTE. En comparación con la clasificación nominal, la ordinal ha sido un campo muy poco explorado dentro del área de aprendizaje automático. Sin embargo, existen varios trabajos y publicaciones relacionados con la regresión ordinal, lo que hace necesario realizar un análisis de los mismos. Principalmente, parece necesario proponer una taxonomía de los métodos existentes, así como realizar el esfuerzo de recopilar las principales métricas de específicas de evaluación. Estos dos temas ayudarán a contextualizar y evaluar las propuestas que se hagan en la materia.

DESBALANCEO DE CLASES. Considerando el obvio carácter multi-clase de los problemas de clasificación ordinal, y que en muchas ocasiones las clases representan grados, este tipo de conjuntos de datos presentan un alto grado de desbalanceo, es decir, algunas clases tienen muy pocos patrones comparadas con el resto de las clases, lo que puede provocar que algunos clasificadores ignoren a las clases con un número significativamente menor de patrones, convirtiéndolos en clasificadores triviales para las clases mayoritarias.

EXPLOTACIÓN DE LA RELACIÓN DE ORDEN DE LAS CLASES. Varios autores definen a los clasificadores ordinales y sus algoritmos de entrenamiento como a) métodos que optimizan la clasificación de acuerdo a métricas que consideren el orden y magnitud de los errores y b) métodos que explotan el conocimiento a priori de la disposición ordenada de los patrones en el espacio de entrada. No obstante, el segundo aspecto no suele contemplarse de manera explícita en la formulación de los clasificadores.

Por último, debemos destacar que consideramos que no sólo es necesaria la propuesta de nuevos métodos, sino también la aplicación de estos nuevos modelos y algoritmos a problemas reales[4]. Así, se identificarán algunos problemas reales de regresión ordinal y serán abordados con las técnicas desarrolladas.

0.5 OBJETIVOS

Esta tesis se centra en investigación en torno a los retos mencionados previamente, aunque los dos últimos serán los principales. El primer reto es un punto de partida necesario, mientras que el segundo no es un problema exclusivo de la clasificación ordinal. Todos estos retos los podemos formalizar en una serie de objetivos que serán abordados en diferentes capítulos:

1. Objetivos para la realización del estado del arte en regresión ordinal (RO):
   a) *Proponer una taxonomía para métodos de RO*. La primera parte en el estudio será la revisión de estado del arte para regresión ordinal. Aunque este objetivo está implícito en cualquier tesis, la RO es un campo relativamente reciente y, hasta donde sabemos, no existen trabajos de revisión específicos para este área. Por tanto, resulta especialmente importante recopilar trabajos existentes, así como proponer una taxonomía que permita organizar los métodos y contribuir con esto al estado del arte. Esta es la razón por la que pensamos que es necesario formalizar este objetivo.
   b) *Recopilar métricas de evaluación de RO*. Tal y como se ha señalado, la RO necesita métricas de rendimiento específicas que no sólo consideren el número de errores, sino también la magnitud de los errores. Es necesario, por tanto, un esfuerzo para identificar todas las propuestas al respecto.
   c) *Seleccionar bases de datos de pruebas*. Una exploración preliminar del estado del arte sugiere que no existen repositorios de bases de datos específicos que sean públicos. El repositorio de bases de datos más utilizado en la literatura es el *ordinal regression benchmark dataset repository* proporcionado por Chu y Ghahra-mani [41]. Sin embargo, estas bases de datos de prueba no representan problemas reales de clasificación ordinal, sino que son problemas de regresión cuya variable de respuesta ha sido discretizada. Respecto a esto, identificamos dos problemas: primero, las bases de datos no pertenecen a problemas de clasificación reales, y cuestiones como el etiquetado erróneo de los patrones o el desbalanceo de clases relacionado con la naturaleza del problema no están presentes; en segundo lugar, al utilizar intervalos de igual ancho para la generación de las etiquetas de clases se produce un desbalanceo artificial de las clases en el conjunto de datos.
2. En torno al problema del desbalanceo de clases se formalizan los siguientes objetivos:

4 De hecho, el grupo de investigación AYRNA, al que el doctorando pertenece, tiene una demostrada experiencia en la aplicación de técnicas de aprendizaje automático a problemas reales, ver publicaciones del grupo en http://www.uco.es/ayrna.

a) *Realizar un análisis del estado del arte para el desbalanceo de clases en clasificación nominal*. Los problemas derivados del desbalanceo de clases han atraído de manera notable la atención de muchos científicos, sobre todo en la última década. Así pues es casi obligatorio explorar brevemente este trabajo previo antes de realizar nuevas propuestas.

b) *Optimizar algoritmos que afronten el problema del desbalanceo de clases nominal como un problema de optimización multi-objetivo*. Una forma de tratar el desbalanceo es mediante algoritmos evolutivos multi-objetivo, tal y como se propone en los trabajos de Fernández-Caballero et al. [21] y Gutiérrez et al. [22]. Los resultados de estos trabajos presentan buena clasificación para todas las clases en entornos multi-clase, aunque su coste computacional es alto. Esto motiva la exploración de alternativas más eficientes, como por ejemplo la familia de algoritmos *extreme learning machine* (ELM) [42], un tipo de algoritmo no iterativo que resulta muy eficiente para el entrenamiento de redes neuronales artificiales.

c) *Explorar nuevas soluciones considerando el desbalanceo de clases ordinal*. Tal y como se ha expuesto, los problemas de RO suelen presentar desbalanceo entre las clases. Los dos objetivos previos servirán para el desarrollo de un nuevo método que trate el problema del desbalanceo en el contexto de la clasificación ordinal.

3. Explotación del orden de los datos. En este trabajo se desarrollarán varios modelos de variable latente, con especial atención a nuevos clasificadores ordinales que exploten mejor el orden de los datos, en torno a esto se formulan los siguientes objetivos:

a) *Comprobar si la explotación del orden de los datos mejora el rendimiento de clasificación en problemas de RO*. Aunque la principal premisa de partida es que en problemas de naturaleza ordinal la RO debería mejorar a la clasificación nominal, los resultados experimentales deberán confirmarla y así, en general, los métodos de RO deberían obtener mejores resultados que los nominales.

b) *Diseñar algoritmos de RO basados en regresión estándar evitando asunciones triviales sobre la variable latente*. Tal y como se ha presentado en este capítulo de introducción, algunas propuestas sugieren simplificaciones que implican tratar el problema como un problema de regresión estándar [25]. Sin embargo, obtener un valor óptimo para representar cada clase es un problema abierto que depende, en general, del problema de clasificación abordado. Uno de los objetivos de esta tesis es extender esta propuesta previa pero evitando simplificaciones triviales sobre las etiquetas de clase.

c) *Desarrollar modelos de variable latente sólo considerando restricciones en el conjunto de etiquetas*. La definición estricta de RO se ajusta únicamente a la restricción en el espacio de la variable objetivo. Por tanto, algunas de las propuestas se desarrollarán considerando sólo esta restricción.

d) *Desarrollar clasificadores que exploten el orden de los datos de entrada*. A pesar de la restricción estricta de RO (ver discusión en el Apartado 3.2 del Capítulo 3), algunos autores como Hühn y Hüllermeier [17] han extendido esta

definición sugiriendo que el orden de las etiquetas puede estar presente en el espacio de entradas. Así pues, tanto los algoritmos de aprendizaje como los modelos podrían beneficiarse de la explotación de esta asunción a priori, y por tanto nosotros estudiaremos esta cuestión con propuestas específicas.

e) *Desarrollar un método que relaje la proyección interna de los modelos de umbral*. Tal y como el lector podrá comprobar en el Capítulo 3, la mayoría de los métodos construyen modelos de proyección que proyectan los patrones en el espacio latente tratando, simultáneamente, de maximizar la distancia inter-clase y minimizar la distancia intra-clase, de esta forma los patrones de la misma clase se proyectan cerca en el espacio latente, y los patrones de las clases diferentes son proyectados lo más separadamente posible. Tal y como se ha comentado en esta introducción, en contextos de solapamiento entre clases o ruido esta filosofía puede generar modelos que impongan transformaciones demasiado rígidas que generalicen peor. Pensamos que relajar estas condiciones de distancias intra e inter clases podría ayudar a mejorar el rendimiento con datos de generalización, al mismo tiempo que supondría una novedad en el modelado de la variable latente.

4. Aplicación de métodos de RO a problemas reales. En esta tesis se afrontarán algunos problemas reales bajo el prisma de la RO con el objetivo último es justificar la investigación en el área. A continuación formulamos los objetivos asociados a dos problemas reales:

   a) *Desarrollo de un sistema de calificación del crédito de los países utilizando regresión ordinal*. La importancia del problema de la calificación del crédito de los estados ha ido creciendo desde el estallido de la crisis financiera. Sin embargo, el papel de las agencias de calificación de crédito en la crisis financiera, entre otros factores, ha motivado una búsqueda de alternativas muy activa en este tipo de problemas. La evaluación de la solvencia de los emisores de deuda normalmente se hace dentro de una escala ordinal. A pesar de esto, las técnicas de RO apenas se han utilizado para evaluar estos problemas, siendo la clasificación binaria el paradigma predominante en los ámbitos financiero y de crédito [43].

   b) *Desarrollo de sistemas de predicción de la velocidad del viento utilizando regresión ordinal*. Los trabajos previos sobre predicción de la velocidad del viento tratan la velocidad del viento como una variable continua. Sin embargo, los gestores de parques eólicos, más que una cifra de velocidad exacta, necesitan tener una idea general de niveles de velocidad determinados por la producción de energía asociada a estos intervalos. Con esta información, los gestores de los parques eólicos pueden optimizar las operaciones de los mismos, por ejemplo, la programación del encendido y apagado de las turbinas de forma óptima. Parece pues interesante estudiar este problema de predicción bajo el enfoque de la regresión ordinal.

0.6 ESTRUCTURA DE LA TESIS

Esta memoria de tesis se organiza de la siguiente forma (al principio de cada capítulo se incluye un resumen de mismo y se enumeran las publicaciones asociadas):

- El Capítulo 1: INTRODUCTION, MOTIVATION AND OBJECTIVES realiza la introducción la tesis en inglés, donde se incluye la presentación del aprendizaje automático y la regresión ordinal y la formalización de objetivos.
- El Capítulo 2: COMPUTATIONAL INTELLIGENCE FOR CLASSIFICATION AND REGRESSION introduce algunos conceptos y técnicas de inteligencia computacional que se emplean o extienden en esta tesis.
- El Capítulo 3: ORDINAL REGRESSION presenta un estudio sobre la regresión ordinal, incluyendo una propuesta de taxonomía para los métodos existentes y un conjunto de métricas de rendimiento específicas para RO. Este capítulo cubre los objetivos 1*a* y 1*b*.
- Capítulo 4: NEW PROPOSALS FOR CLASS IMBALANCE PROBLEM presenta una revisión de literatura relacionada con el desbalanceo de clases, y además muestra nuevas propuestas para entrenamiento de clasificadores sensibles al problema del desbalanceo de clases, con especial énfasis en la optimización del coste computacional. Este capítulo cubre los objetivos 2*a* y 2*b*.
- El Capítulo 5: NEW PROPOSALS FOR ORDINAL REGRESSION recoge distintas propuestas específicas para RO. La primera propuesta presentada en este capítulo sirve de enlace entre los trabajos de desbalanceo de clases y la regresión ordinal (objetivo 2*c*). La segunda propuesta enfoca el problema de RO como un problema de regresión estándar (objetivo 3*b*) tanto con explotación del orden de etiquetas (objetivo 3*c*) como explotación del orden de los datos de entrada (objetivo 3*d*). La última propuesta de este capítulo propone una proyección del espacio de entrada de los datos al espacio latente que recoge la idea de relajar la proyección tal y como identifica el objetivo 3*e*. De forma transversal, se recogen varias bases de datos de RO, de manera que el capítulo también contempla el objetivo 1*c*.
- El Capítulo 6: APPLICATION OF ORDINAL REGRESSION TO SOVEREIGN CREDIT RATING presenta la aplicación del método PCDOC (propuesto en el Capítulo 5) al problema de la evaluación del crédito de los países. En este capítulo no sólo se presenta un sistema de clasificación, sino también se propone el uso de la proyección realizada por el regresor asociado al método PCDOC como técnica de visualización del rango de los países que puede ser incorporada a sistemas de apoyo a la decisión. Este capítulo cubre el objetivo 4*a*.
- El Capítulo 7: APPLICATION OF ORDINAL REGRESSION TO WIND SPEED FORECASTING describe la aplicación de técnicas de RO a la predicción de velocidad del viento como herramienta de ayuda a los gestores de parques eólicos. Este capítulo cubre el objetivo 4*b*.

- El Capítulo 0: RESUMEN DE LA TESIS finaliza la memoria de la tesis con un resumen de las contribuciones científicas y proporciona algunas pistas sobre posibles líneas de investigación futuras.

Por último, los resultados experimentales de los capítulos 5, 6 y 7 confirman la hipótesis planteada en el objetivo 3*a* porque se muestra, en general, una mejora del rendimiento obtenido por los métodos de RO frente a métodos nominales similares.

0.7 RESUMEN Y CONCLUSIONES

Esta memoria de tesis presenta un trabajo de investigación en el campo de la regresión ordinal orientada fundamentalmente en torno a dos problemas: el desbalanceo de clases – problema común con otros tipos de clasificación – y la explotación del orden de los patrones de las clases con el fin de mejorar el rendimiento de los clasificadores. En este apartado resumiremos las principales contribuciones de la tesis agrupadas por temas.

0.7.1 *Revisión de trabajos y taxonomía de métodos*

La contribución científica de la tesis comienza en el Capítulo 3, donde se realiza un estudio exhaustivo de métodos de regresión ordinal. En el momento de escribir esta tesis, no tenemos conocimiento de trabajos de revisión similares para esta materia. El capítulo define formalmente el problema de la regresión ordinal, y lo diferencia claramente de otros problemas relacionados. Tras esto se presenta una propuesta de taxonomía para métodos de RO que los divide en cuatro grandes grupos: aproximaciones ingenuas o simplistas (*naïve approaches*), descomposiciones binarias (*binary decompositions*), modelos de umbral (*threshold models*) y clasificación binaria aumentada (*augmented binary classification*).

En nuestra opinión, la taxonomía propuesta puede ayudar a futuros investigadores a proponer y desarrollar métodos de RO, permitiendo categorizar las nuevas propuestas y a analizar los métodos similares.

0.7.2 *Desbalanceo de clases*

El Capítulo 4 está dedicado al primero de los dos grandes objetivos de investigación de la tesis: el problema del *desbalanceo de clases* (también denominadas *bases de datos desbalanceadas*). Tal y como se ha introducido, en los últimos años, y especialmente dentro de área de clasificación nominal, el tema del desbalanceo de clases ha motivado mucha actividad en torno al diseño de clasificadores que consideren todas las clases del problema, así como al desarrollo de métricas de evaluación específicas que consideren el rendimiento de clasificación clase a clase.

No obstante, el problema del desbalanceo de clases se ha afrontado fundamentalmente para problemas de clasificación binaria, debido a una serie de barreras que evitan aplicar técnicas robustas, como el análisis ROC, a entornos multi-clase. Recientemente, este tipo de problemas desbalanceados multi-clase se han abordado desde un enfoque de optimización multi-objetivo. Sin embargo, el coste computacional de

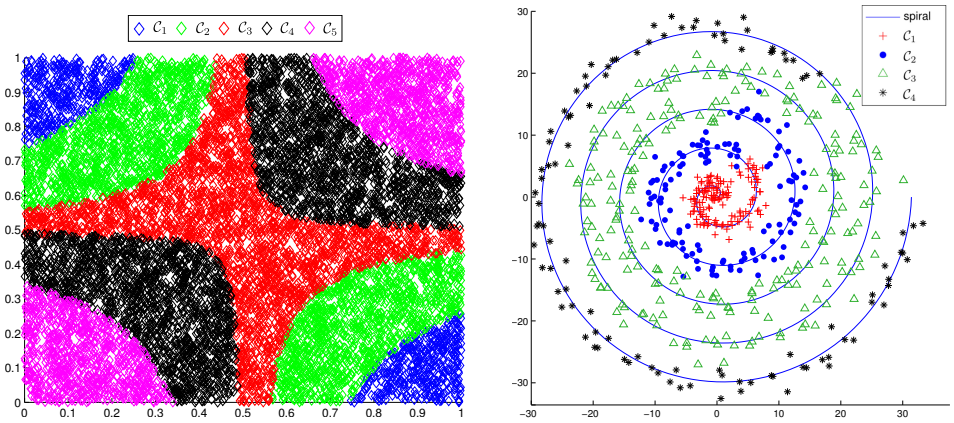
estas propuestas motiva nuevos trabajos de investigación para obtener métodos más eficientes.

Así, en el Capítulo 4 se exploran varias alternativas para afrontar el problema del desbalanceo en entornos multi-clase de una manera más eficiente. Considerando el rendimiento en clasificación para todas las clases y el tiempo computacional, se seleccionó como mejor opción la basada en una salida del modelo de red continua y probabilística evaluada mediante una función basada en la raíz del error cuadrático medio ($RMSE$). Los puntos que hacen efectiva y eficiente la alternativa seleccionada son tres. Primero, las anteriores propuestas, basadas en frentes de Pareto, son reformuladas como un problema de optimización convexo de una combinación lineal y ponderada de los objetivos considerados. Segundo, las funciones de error (a partir de las salidas del modelo) se diseñaron para producir respuestas continuas y probabilísticas. En este sentido se presentaron varias propuestas para guiar un algoritmo evolutivo, destacando dos de las funciones de error continuas, una basada en la entropía cruzada y la otra basada en el $RMSE$. La segunda resultó más robusta y más fácil de calcular en términos de coste computacional. Estos dos factores redujeron notablemente el coste de la evaluación de las soluciones candidatas, a la vez que ayudaron a generar clasificadores más robustos. El tercer factor clave en el diseño de este sistema eficiente fue la selección del algoritmo *Evolutionary Extreme Learning Machine (E-ELM)*, un algoritmo muy eficiente para el entrenamiento de redes neuronales.

0.7.3 *Modelos de regresión ordinal y aprendizaje*

El segundo gran tema de investigación de esta tesis es la exploración de nuevos modelos y algoritmos de aprendizaje para regresión ordinal. Las contribuciones en esta materia se hacen en el Capítulo 5.

La primera propuesta es el algoritmo E-ELMOR, que hereda algunas de las lecciones aprendidas en el Capítulo 4. En este sentido, se propone la función de error $RMSE$ ponderado (*Weighed RMSE*, $WRMSE$) como propuesta para guiar la búsqueda de soluciones candidatas en un algoritmo evolutivo. El $WRMSE$ refleja de forma simultánea tres criterios que son deseables en un clasificador ordinal, a la vez que tiene en cuenta el problema del desbalanceo de clases: a) los errores de clasificación de clases no adyacentes deben ser más penalizados según la distancia entre las clases crece; b) la probabilidad a posteriori de pertenencia de un patrón a cada clase debe ser unimodal y decrecer monótonamente hacia las clases no adyacentes; c) en otras métricas de error, como el MZE (*Mean Zero-one Error* o ratio de error medio), sólo una de las salidas de la red neuronal (la que tenga valor máximo) suele contribuir a la función de error, y además no contribuye con el valor de salida, por el contrario, en métricas basadas en el $RMSE$, y teniendo disponible una salida probabilística continua, cada salida del modelo (probabilidades a posteriori) contribuye a la función de error de manera que los umbrales de salida y probabilidades a posteriori tenderán a ser más discriminantes. Todo esto es posible porque en este trabajo utilizamos un único modelo multi-clase, ya que estas ideas no serían directamente trasladables a esquemas como las descomposiciones binarias. Los resultados de *Evolutionary Extreme Learning Machine for Ordinal Regression* (E-ELMOR) suponen una mejora considerable respecto a los métodos de referencia.



(a) Representación de la base de datos *toy*.

(b) Representación de la base de datos *spiral*.

Figura 0.5: Representaciones de las bases de datos *toy* (propuesta por Herbrich et al. [8]) y la base de datos *spiral* (propuesta en el Capítulo 5).

El resto de propuestas se centran en transformar el problema de clasificación en un problema de regresión estándar como forma de modelar la variable latente desconocida –que refleja la disposición de orden entre los patrones de las diferentes clases–.

En esta línea, la segunda propuesta es el método *Numerical Variable Reconstruction* (NVR). Esta alternativa se ciñe a la definición estricta de RO en la que la restricción de orden sólo se aplica a las etiquetas, esto es, al espacio de salida. Este método ajusta una serie de distribuciones de probabilidad para muestrear valores continuos que se utilizan para representar a cada patrón en el espacio latente. Dependiendo de la clase, el valor latente correspondiente al patrón se extrae de una función de probabilidad diferente, de manera que los valores latentes para los patrones de una misma clase siempre están acotados en el mismo intervalo. Una vez obtenida la representación latente de los patrones, se entrena un modelo de regresión para predecir esta variable. NVR funciona de manera aceptable en algunas bases de datos, sin embargo, cuando extendemos la experimentación a más bases de datos y utilizamos más métricas de evaluación del rendimiento, el método no resulta suficientemente robusto.

En este punto, consideramos que la idea de producir una variable continua asociada a los patrones y entrenar un regresor para predecir esta variable podría seguir siendo válida, pero esta variable debería ser generada de una forma más adecuada. A pesar de que la definición estricta de RO no considera orden en el espacio de entrada, los modelos de umbral asumen de manera implícita que el espacio latente refleja de alguna forma el orden total de los patrones. Así pues, surge la idea de explotar de forma explícita este orden para construir la variable latente.

Para esto, nuestro objetivo es capturar el orden de los datos en la proyección unidimensional que es el espacio latente, de manera que las posiciones relativas de los patrones en el espacio de entrada se trasladen a posiciones relativas en el espacio latente. Un idea inicial podría ser, por ejemplo, utilizar el centroide de cada clase, de manera que los patrones se posicionen en el espacio latente según su distancia al centroide de la clase. Sin embargo, una observación inicial de los conjuntos de datos de ejemplo de la Figura 0.5 sugiere que esto no es un idea robusta, porque en estos

conjuntos de datos los centroides de las clases son muy similares. Así surge la idea de utilizar distancias entre pares de clases vecinas para realizar la proyección de los patrones al espacio unidimensional. De esta forma nos ceñimos a la definición de orden de la RO donde la relación orden está garantizada en las clases vecinas.

La idea expresada en el anterior párrafo es implementada por la propuesta de proyección de *distancias entre pares de clases* (*Pairwise Class Distance* (PCD)), y el clasificador asociado, denominado *proyección basada en distancias entre pares de clases para clasificación ordinal* (*Pairwise Class Distances for Ordinal Classification* (PCDOC)). Experimentalmente, concluimos que el método PCDOC alcanzó un rendimiento adecuado en comparación con varios métodos de estado del arte. Además, dentro de los experimentos estudiamos en profundidad las diferentes formas de modelado de los métodos de umbral. Tras este análisis concluimos que el resto de métodos tienen a compactar las proyecciones de los patrones de la misma clase, haciendo que el valor latente de los patrones de cada clase (en el conjunto de entrenamiento) sea prácticamente el mismo. En algunas bases de datos, esta proyección tan forzada a través de transformaciones altamente no lineales puede derivar en un peor rendimiento de generalización. Esta forma de proyección es más relajada en el método PCDOC, que tiende a producir modelos de proyección más suaves.

En conclusión, los resultados indican que nuestra propuesta de dos fases para la clasificación ordinal es una alternativa a los métodos del estado del arte que resulta viable y fácil de entender. Además, en el caso de PCDOC, la proyección construida en la primera fase extrae de forma consistente información útil para la clasificación. Un ejemplo de la utilidad de esta información es la aplicación de esta técnica al problema de la evaluación del crédito soberano, realizada en el Capítulo 6, y que se resume en el siguiente apartado.

0.7.4 *Mejora de problemas de aplicación reales bajo el enfoque de regresión ordinal*

Dos problemas de aplicación reales se han presentado en esta tesis.

La primera aplicación es abordada en el Capítulo 6, y consiste en la calificación de crédito de los países utilizando el método PCDOC, el cuál es comparado experimentalmente con otros clasificadores nominales y ordinales. La robustez del método PCDOC, así como la de otros métodos, es destacable considerando varias métricas de evaluación.

Además de realizar la tarea de clasificación, la proyección del modelo de regresión interno del método PCDOC se utilizó como técnica de visualización de la posición de los países evaluados, lo que proporciona una buena herramienta para construir, por ejemplo, sistemas de ayuda a la decisión. En comparación con otras técnicas de visualización de datos no supervisadas, la proyección del modelo PCDOC está validada a través de su idoneidad para clasificar patrones de manera correcta.

La segunda aplicación es un problema de predicción de la velocidad del viento, y se trata en el Capítulo 7. En este capítulo proponemos transformar los valores continuos de velocidad del viento en un conjunto de etiquetas ordinales, de forma que cada etiqueta está relacionada con la energía que se puede generar según el rango de valores de velocidad del viento. En el capítulo se realizan una serie de experimentos extensivos, comparando la capacidad de varios clasificadores nominales y ordinales para predecir la etiqueta.

La conclusión más importante de estos dos últimos capítulos es que el abordar estos dos problemas como técnicas de regresión ordinal mejoró el rendimiento de generalización. Esto, en definitiva, justifica la presente tesis y motiva futuras investigaciones en el campo de la regresión ordinal.

0.8 TRABAJO FUTURO

Hay una serie de líneas de investigación que podrían extender el trabajo presentado en esta tesis.

0.8.1 *Posibles mejoras para los métodos propuestos*

En relación a los métodos propuestos para regresión ordinal, las siguientes cuestiones podrían ser exploradas:

- En el caso de E-ELMOR, el trabajo futuro podría implicar el diseño y experimentación con nuevos códigos de salida y nuevas funciones de error asociadas. Por ejemplo, se podrían diseñar nuevos códigos de salida para la red neuronal que consideren la distancia entre las etiquetas de clase.
- En relación a la proyección PCD, al final del Capítulo 5 se realiza una discusión sobre la posible influencia no deseada de valores atípicos o *outliers* en la proyección. Tal y como se sugiere en el propio capítulo, una alternativa directa sería utilizar un esquema del tipo k-NN para calcular la distancia mínima de un patrón a los patrones de las clases vecinas. De esta forma, en lugar de utilizar el valor mínimo de las distancias a puntos de una clase, se puede utilizar la media de los k valores mínimos de las distancias a puntos de las clases $q \pm 1$. Esto supondría una generalización de la propuesta que ahora mismo calcula estas distancias con un valor de $k = 1$. Sin embargo, la inclusión de k implica añadir un nuevo parámetro libre al proceso de entrenamiento.

0.8.2 *La cuestión de la evaluación del orden de los datos*

Distintos experimentos realizados durante el desarrollo de esta tesis han revelado que algunos clasificadores nominales obtenían mejor rendimiento que los clasificadores ordinales en algunas bases de datos *aparentemente* ordinales. Esto nos lleva a pensar que incluso cuando la naturaleza de un problema sugiere que existe una relación de orden entre las clases, este orden puede que no esté reflejado en el espacio de entrada.

Así, como punto de partida para una posible línea de investigación, en este apartado realizamos una serie de experimentos para alterar de forma artificial el orden de las clases. El objetivo es comprobar cuándo se ve alterado el rendimiento de los clasificadores ordinales si se cambia la restricción de orden. Para estos experimentos, simplemente reetiquetamos las bases de datos para alterar el orden inicial de las clases y por tanto el orden relativo de los patrones en el espacio de entrada. El reetiquetado consiste en permutaciones de las etiquetas de clase. Para cada base de datos en cuestión se realiza una serie de experimentos con las etiquetas originales

(*Original*), una permutación aleatoria de las etiquetas (*Shuffle*), y un orden inverso de las etiquetas (*Inverse*). La Tabla 0.1 muestra las tres combinaciones de etiquetas utilizadas en los experimentos dependiendo del número de clases del problema y la Tabla 0.2 muestra las características de las diferentes bases de datos utilizadas para estos experimentos.

Tabla 0.1: Etiquetado original y opciones de reetiquetado (etiquetas «aleatorias» (*Shuffle*) y orden inverso de las etiquetas (*Inverse*)). Observe que la segunda opción no es una ejecución puramente aleatoria sino que el orden es alterado con la restricción de evitar órdenes parciales entre las clases, y además, la combinación «aleatoria» resultante es la misma para todos los experimentos para poder realizar comparaciones entre métodos.

Número de clases	*Original*	*Shuffle*	*Inverse*
2	[1,2]	[2,1]	[2,1]
3	[1,2,3]	[1,3,2]	[3,2,1]
4	[1,2,3,4]	[1,4,2,3]	[4,3,2,1]
5	[1,2,3,4,5]	[3,1,5,2,4]	[5,4,3,2,1]
6	[1,2,3,4,5,6]	[1,5,2,4,6,3]	[6,5,4,3,2,1]
7	[1,2,3,4,5,6,7]	[3,1,7,5,2,6,4]	[7,6,5,4,3,2,1]
8	[1,2,3,4,5,6,7,8]	[3,8,1,7,5,2,6,4]	[8,7,6,5,4,3,2,1]
9	[1,2,3,4,5,6,7,8,9]	[3,8,1,7,5,9,2,6,4]	[9,8,7,6,5,4,3,2,1]
10	[1,2,3,4,5,6,7,8,9,10]	[3,10,8,1,7,5,9,2,6,4]	[10,9,8,7,6,5,4,3,2,1]

Los experimentos se realizaron de una forma similar a como se han hecho en el resto de la tesis, en este caso la precisión (*Acc*) es la métrica utilizada para la comparación. No utilizamos métricas de RO ya que en estos experimentos no se asume ninguna relación de orden concreta entre las etiquetas. Por este motivo *Acc* es la métrica utilizada también como criterio de selección de los hiper-parámetros de los métodos.

La Figura 0.6 muestra el rendimiento de varios clasificadores en las mismas bases de datos con distinto etiquetado. La figura revela que varios de los métodos de RO tienen una caída notable en el rendimiento cuando el orden de las etiquetas es alterado (y por tanto el orden de los datos en el espacio de entrada). En especial, los métodos de umbral (PCDOC, *Support Vector Ordinal Regression with implicit constraints* (SVORIM) y *Kernel Discriminant Learning for Ordinal Regression* (KDLOR)) se ven afectados muy negativamente por el etiquetado aleatorio. En el caso de PCDOC y KDLOR, el rendimiento también se degrada para el caso de la inversión de etiquetas, mientras que el método SVORIM no se vio afectado. El método *Extreme Learning Machine for Ordinal Regression* (ELMOR) también está afectado por la caída del rendimiento, pero la pérdida de rendimiento relativa es menor. Como era de prever, el método *Cost Support Vector Classification* (SVC) no está afectado por el reetiquetado de las bases de datos[5]. De este estudio surgen las siguientes preguntas: a) ¿por qué la

5 Cabe destacar que, aunque SVC es un método determinista, hay pequeñas variaciones en el rendimiento del clasificador. El motivo es que la selección de hiper-parámetros se realiza mediante una búsqueda en

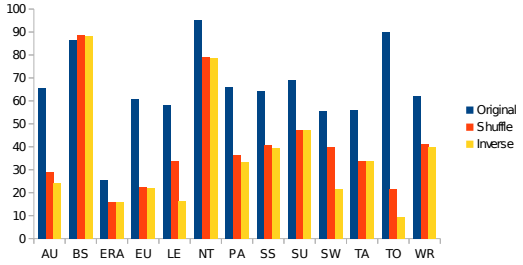
Tabla 0.2: Características de las bases de datos de prueba consideradas para los experimentos de reetiquetado

Base de datos	Nº Patr.	Nº Atri.	Nº Clases	Distribución de las clases
automobile (AU)	205	71	6	$(3, 22, 67, 54, 32, 27)$
balance-scale (BS)	625	4	3	$(288, 49, 288)$
ERA (ER)	1000	4	9	$(92, 142, 181, 172,$ $158, 118, 88, 31, 18)$
eucalyptus (EU)	736	91	5	$(180, 107, 130, 214, 105)$
LEV (LE)	1000	4	5	$(93, 280, 403, 197, 27)$
newthyroid (NT)	215	5	3	$(30, 150, 35)$
pasture (PA)	36	25	3	$(12, 12, 12)$
squash-stored (SS)	52	51	3	$(23, 21, 8)$
squash-unstored (SU)	52	52	3	$(24, 24, 4)$
SWD (SW)	1000	10	4	$(32, 352, 399, 217)$
tae (TA)	151	54	3	$(49, 50, 52)$
toy (TO)	300	2	5	$(35, 87, 79, 68, 31)$
winequality-red (WR)	1599	11	6	$(10, 53, 681, 638, 199, 18)$

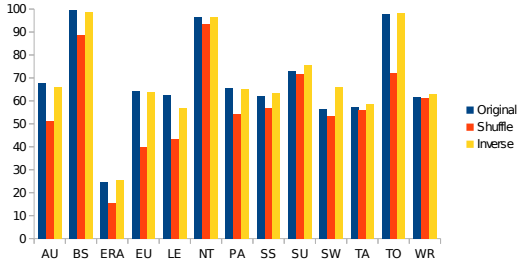
caída de rendimiento es mayor para algunas bases de datos? b) ¿por qué algunos de los métodos de RO son más robustos a este proceso de reetiquetado?. La respuesta a la pregunta a) está relacionada con el hecho de que la estructura ordinal puede encontrarse en el espacio de etiquetas o espacio de salida, pero no en el espacio de entrada. La respuesta a la pregunta b) debería encontrarse en la forma en que cada algoritmo construye y optimiza los diferentes clasificadores.

En conclusión, y considerando estos experimentos preliminares, existe motivación suficiente para realizar un análisis del espacio de entrada con el fin de evaluar el grado de orden de los problemas. Así, una de las líneas de trabajo futuro debería incluir el desarrollo de métodos para evaluar lo oportuno de abordar un problema como una tarea de clasificación ordinal o como una de clasificación nominal en función de evaluaciones previas del conjunto de datos.

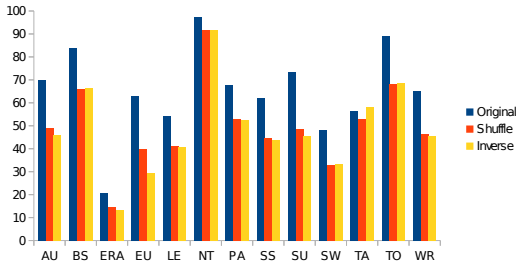
malla que realiza un proceso de validación cruzada interno. Este proceso de validación realiza diferentes particiones de los datos de entrenamiento, produciendo conjuntos de entrenamiento y validación que dependen de una semilla aleatoria. Especialmente para conjuntos de datos pequeños, la selección de algunos de los patrones para los conjuntos de entrenamiento y validación puede influir en la selección final de hiper-parámetros, y en consecuencia el rendimiento en generalización se ve afectado.



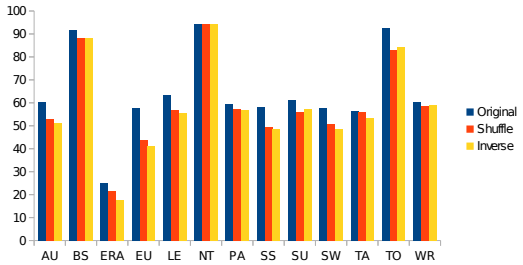
(a) SVRPCDOC.



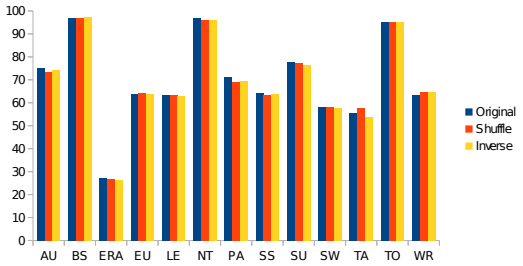
(b) SVORIM.



(c) KDLOR.



(d) ELMOR.



(e) SVC.

Figura 0.6: Experimentos preliminares que muestran el rendimiento en *Acc* de diferentes métodos con las etiquetas originales (*Original*), etiquetas «aleatorias» (*Shuffle*) y etiquetas con orden inverso (*Inverse*).

Part I

INTRODUCTION

This first part of the thesis presents the ordinal regression problem and contextualizes it in the field of machine learning and computational intelligence. Chapter 1 presents a general introduction to machine learning and ordinal regression in order to state the objectives of the thesis. Chapter 2 presents the main machine learning paradigms related to this thesis. In Chapter 3, a state-of-the-art review regarding ordinal regression is developed.

1

INTRODUCTION, MOTIVATION AND OBJECTIVES

Summary. This chapter introduces the context of machine learning field to study the ordinal classification problem. Ordinal classification lies in the area of supervised classification techniques, also referred to as "classifiers", "learning algorithms", or simply "learners". A brief introduction to supervised learning and ordinal regression is done in order to contextualize the reader and state the main objectives. The chapter ends up with a road map of the thesis dissertation.

1.1 MACHINE LEARNING AND SUPERVISED LEARNING

Machine learning does not have a well established definition as pointed out by Professor Andrew Ng [10]. In 1959, Arthur Samuel defined machine learning as a "Field of study that gives computers the ability to learn without being explicitly programmed" [10, 11]. Later, Tom M. Mitchell stated "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" [12]. Then, machine learning is equivalent to "learning from data" in order to extract knowledge for several purposes, for instance to help humans to take decisions or even automate decisions from data, as well as adapting systems dynamically to enable better user experiences [13]. This "learning from data" makes machine learning to be placed somewhere between different science fields belonging to Artificial Intelligence, Statistics and Mathematics (see Figure 1.1). Depending of how the learning is done, different disciplines are developed, being provably the most active the computational intelligence [14], which encloses artificial neural networks [15, 16], fuzzy systems and evolutionary computation, including swarm intelligence. For historical reasons some statistical and mathematical techniques, such as Support Vector Machines, rough sets, Bayesian networks and probabilistic reasoning, or even statistical natural language processing, are accepted as valid computational intelligence topics by computational intelligence conferences and journals, while many others, such as numerical optimizations techniques, approximation theory, statistical methods or first-order logic are beyond their scope [14].

One of the most active fields in machine learning is *pattern recognition*, that broadly speaking can be divided into *supervised learning*, which can be informally described as teaching the computer how to do something, then letting it use the new knowledge found to do it, and *unsupervised learning* in which the computer learn how to do something, and we use this to determine structure and patterns in data. In the first

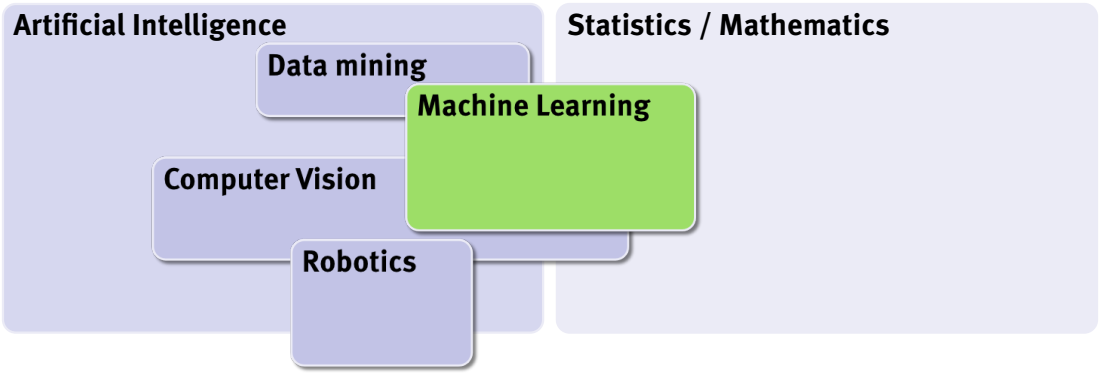
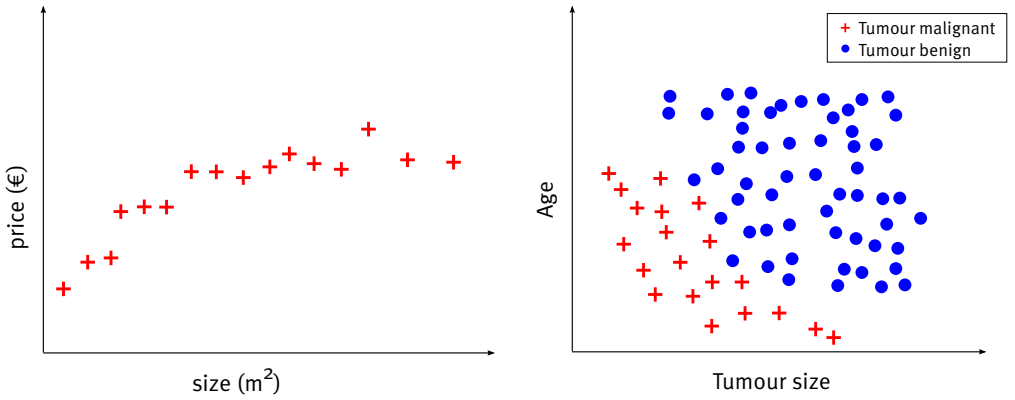


Figure 1.1: Machine learning: Where does it fit? What is it not? (source [13])

case, we provide the computer with *labeled* data whereas, in the second case, the data is *unlabeled* and the computer has to structure it.

Supervised learning is probably the most common problem type in machine learning. For instance, let us suppose we want to predict house prices, see Figure 1.2a. We can collect data regarding housing prices and how they relate to size in square meters, and we can provide the algorithm this *training data* which is composed of the independent variables (the size in m^2) and the dependent variables[1] or "right answers". The learning algorithm has to build a model from this labeled (supervised) training data and it has to be able to predict the right price values, that we do not know, from new (previously unseen) data. In this case, the problems are named to as *regression problems* because the target variable is continuous. The unseen data used for testing the real performance of the predictor is typically called *generalization data*, *test data* or *testing data*.



(a) Example regression problem "Given these data, a friend has a house of 75 square meters, how much can he expect to get?".

(b) Example classification problem "Can you estimate prognosis based on tumor size and known age?"

Figure 1.2: Example of regression and classification problems (source [10]).

On the other hand, when the predicted variable is discrete, the problem is called *classification problem*. For example, Figure 1.2b shows a classification problem in which

1 Typically there will be an unique dependent variable, however several supervised classification branches, such as multi-label classification presents more than one dependent variable.

we try to define breast cancer as malignant or benign based on tumour size and patient's age[2]. Based on the training data, the algorithm should be able to separate the two classes. The output variable can have more than two classes, for instance class benign, malignant type A, malignant type B and malignant type C. The former are referred to as binary problems while the later are multi-class problems. There is a third class of classification problems, which are indeed the main target of this thesis. If there is an relation order between the classes, the problems are called *ordinal classification* problems. In the example, if we want to classify a tumour in classes "benign", "suspiciously malignant", "malignant" and "alarming malignant", we are probably talking about ordinal classification because we are dealing with different degrees of an illness. Ordinal classification is also known as *ordinal regression* because it has both relation to classification and regression, as will be explained in the next section.

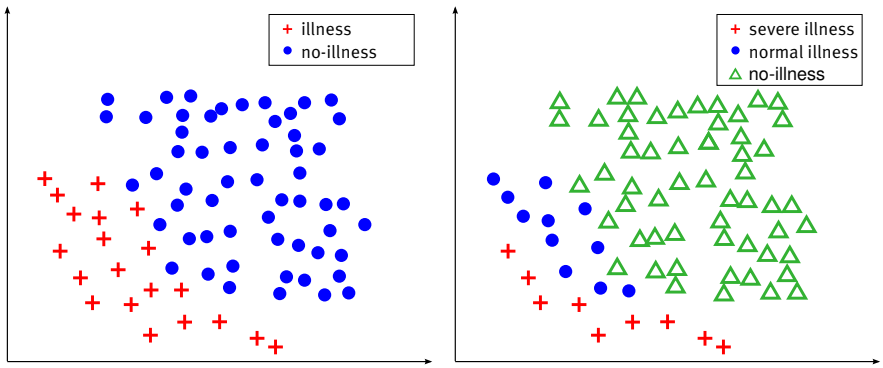


Figure 1.3: An example of a binary (figure on the left side) and an ordinal classification (figure on the right). In the first case, there are two status for a pattern: illness or no-illness. Nevertheless, the expert can demand to have finer classification targets, so the same problem can be addressed as a multi-class problem, which is the second case. In addition, if the classes have an order arrangement between them, then we are dealing with an ordinal regression problem, which is a more precise approach for the above example.

Figure 1.3 shows an example of a binary classification problem compared to an ordinal classification one with the purpose of detecting an illness. In the first case, the learner will only be able to detect presence or absence of an illness. In the second case, more degrees are allowed reflecting the severity of the illness. Up to this point, we can talk about nominal multi-class problems. However a main issue makes ordinal regression different to nominal classification: because of the problem nature, there is an order arrangement between labels, and this order should be also present in the attributes space[3]. This affects the classifier in two ways: first, the classifier should exploit this a priori knowledge about the patterns distribution through the input space [17] and, second, the classifier evaluation should be done with specific performance measures [18]. That is, misclassifying a *severe illness* pattern as *absence* must be more penalized than misclassifying the same pattern as *normal illness*. These

2 These variables have been selected for illustration purpose, it is obvious that a tumour cannot be classified only in terms of those variables.

3 Although it can happens that the problem nature suggests an order arrangement between labels that is not reflected in the input space.

two challenges draw the main differences with nominal classification, and so we can state that addressing these problems as ordinal regression problems can help to achieve the ultimate motivation of machine learning, "*learning from data*".

1.2 ORDINAL CLASSIFICATION

Ordinal classification or ordinal regression (OR) is a supervised learning problem of predicting categories that have an ordered arrangement. When the problem is really exhibiting an ordinal nature, it is expected that this order is also present in the data input space [17]. The samples are labelled by a set of ranks with an ordering amongst the categories. In contrast to nominal classification, there is an ordinal relationship throughout the categories and it is different from regression in that the number of ranks is finite and exact amounts of difference between ranks are not defined. In this way, ordinal classification lies somewhere between nominal classification and regression.

These classification problems should not be confused with sorting or ranking. Sorting is related to ranking all samples in the test set, with a total order. Ranking is related to rank the samples with a relative order, and with a limited number of ranks (partial order). Of course, ordinal regression can be used to rank samples, but its objective is to obtain a good accuracy, and, at the same time, a good ranking.

The relevance of ordinal classification is obvious because this type of problems are common in our everyday life, where many problems require classification of items into naturally ordered classes. Examples of these problems are the teaching assistant evaluation [1], car insurance risk rating [2], pasture production prediction [3], many problems from the field of preference learning [7], breast cancer conservative treatment [4] or credit rating [6].

Ordinal classification problems present two main issues to be taken into account by the learning algorithm. Firstly, the nature of the problem implies that the class order is somehow related to the distribution of patterns in the space of attributes, and also to the topological distribution of the classes (although, in general, this relation will be a nonlinear one). Therefore the classifier must exploit this a priori knowledge about the input space [17]. Secondly, when evaluating an ordinal classifier, the performance metrics must consider the order of the classes so that misclassifications between adjacent classes should be considered less important than the ones between non-adjacent classes, more separated in the ordinal scale. For example, given an ordinal dataset of weather prediction with a target variable taking values in the set $\{Very\ cold, Cold, Mild, Hot, Very\ hot\}$, the natural order between classes, $Very\ cold \prec Cold \prec Mild \prec Hot \prec Very\ hot$ is clear. It is straightforward to think that predicting class *Hot* when the real class is *Cold* represents a more severe error than that associated with a *Very cold* prediction [18]. Thus, specialized measures are needed for evaluating ordinal classifier performance [18–20].

Whereas Accuracy (Acc) (also known as Correct Classification Rate (CCR)) is the most common performance metric for nominal classification, the Mean Absolute Error (MAE) is the most commonly used one in the context of ordinal regression. In multi-class classification, alternative metrics have been proposed, for instance, to measure the performance of individual classes [21, 22] (including those classes which are worse classified). Similarly, the Averaged Mean Absolute Error (AMAE) has being

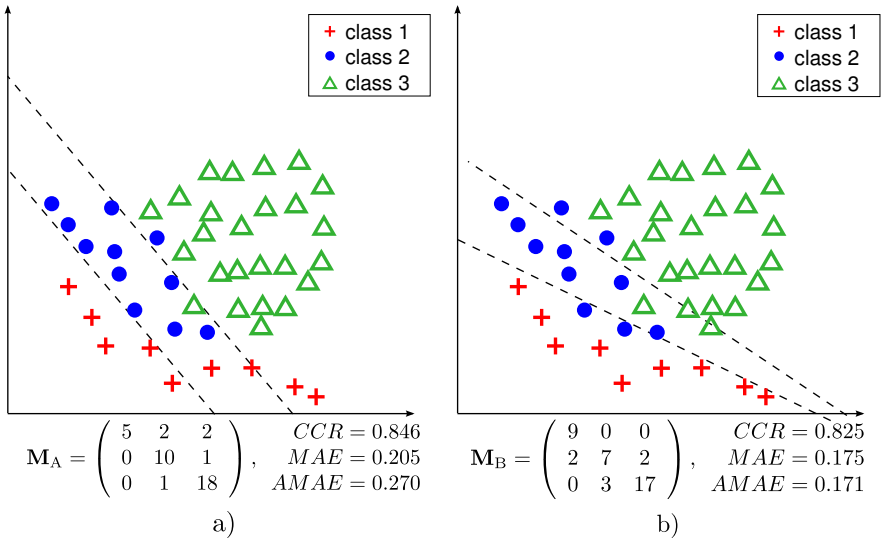


Figure 1.4: Example of the necessity of using alternative performance metrics for ordinal regression. Let us suppose this is the previous illness detection multi-class problem where class 1 is *severe illness*, class 2 means *normal illness* and class 3 means *absence*. In the example, the same point set is classified by two different classifiers a) and b). Classifier a) has the best performance regarding CCR, however classifier b) has the best performance when considering MAE and AMAE metrics. In this case, although a) is having the best classification ratio, it makes relevant mistakes because it is misclassifying pattens of class 1 (*severe illness*) into class 3 (*absence*). Observe that MAE and AMAE metrics improve for classifier b) pointing out that b) is a better ordinal classifier than a) for the current dataset sample.

proposed by Baccianella et al. [23] in order to better evaluate the ordinal classification performance in imbalanced datasets (those classification problems where the number of patterns for each class is very different). Figure 1.4 presents a motivational example of an ordinal classification problem in which we want to compare the performance of two classifiers with the same dataset. In the example, classifier b) is a better ordinal classifier than classifier a), even thought the (global) Accuracy of a) is the highest.

1.3 BRIEF STATE-OF-THE-ART OF ORDINAL CLASSIFICATION

Variety of approaches have been proposed for ordinal classification. For example, Raykar et al. [24] learnt ranking functions in the context of ordinal regression and collaborative filtering datasets. Kramer et al. [25] mapped the ordinal scale by assigning numerical values and then applied a regression tree model.

Other simple alternative that appeared in the literature tried to impose the ordinal structure through the use of cost-sensitive classification, where standard (nominal) classifiers are made aware of ordinal information through penalizing the misclassification errors, usually by selecting a cost equal to the absolute deviation between the actual and the predicted labels [26].

The third direct alternative suggested in the literature is to transform the ordinal classification problem into a nested binary classification one [27, 28], and then to combine the resulting classifier predictions to obtain the final decision. It is clear that

ordinal information allows ranks to be compared. For a given rank k, an associated question could be "is the rank of pattern $\mathbf{x}$ greater than k?". This question is exactly a binary classification problem, and ordinal classification can be solved by approaching each binary classification problem independently and combining the binary outputs to a rank [27].

Other alternative [28] imposed explicit weights over the patterns of each binary system in such a way that errors on training objects were penalized proportionally to the absolute difference between their rank and k. Binarization of ordinal regression problems can also be tackled from augmented binary classification perspective, i.e. binary problems are not solved independently, but a single binary classifier is constructed for all the subproblems. For example, Cardoso and Pinto da Costa [29] added additional dimensions and replicated the data points through what is known as *the data replication method*. This augmented space was used to construct a binary classifier and the projection onto the original one resulted in an ordinal classifier. A very interesting framework in this direction is that proposed by Li and Lin [30], Lin and Li [31], reduction from cost-sensitive ordinal ranking to weighted binary classification (RED), which is able to reformulate the problem as a binary problem by using a matrix for the extension of the original samples, a weighting scheme and a V-shaped cost matrix. An attractive feature of this framework is that it unifies many existing ordinal ranking algorithms, such as perceptron ranking [32] and support vector ordinal regression [33]. Recently, the Learning Vector Quantization (LVQ) was adapted to the ordinal case in the context of prototype based learning [34]. In that work the order information is utilized to select class prototypes to be adapted, and to improve the prototype update process.

Moreover, other proposals addressing ordinal classification – indeed the vast majority of them – can be grouped under the umbrella of *threshold methods* [35]. These methods assume that ordinal response is a coarsely measured latent continuous variable, and model it as real intervals in one dimension. Based on this assumption, the algorithms seek a direction onto which the samples are projected and a set of thresholds that partition the direction into consecutive intervals representing ordinal categories [35–39].

1.4 MOTIVATION AND CHALLENGES

As previously mentioned, ordinal classification deals with supervised classification problems in which there is an order within categories. This order is typically deduced from the problem nature by an expert or by a simple inference about the data. For instance, illness detection example can be considered as an ordinal classification problem with the purpose of assigning the right ordered category to a person given a set of input variables, being the category label set $\{C_1 = \textit{risk}, C_2 = \textit{severe}, C_3 = \textit{normal}, C_4 = \textit{possible presence}, C_5 = \textit{absence}\}$, where labels represent the illness degree assigned by doctors. Here there is a natural order between classes ($\textit{risk} \prec \textit{severe} \prec \textit{normal} \prec \textit{possible presence} \prec \textit{absence}$), *risk* being the worse illness degree and *absence* the best one. In this case, the ordinal nature of the problem can be deduced, not only from the dependent variables, i.e. the labels, but also for the independent variables, that reflect this order in the space of attributes $\mathcal{X}$. For example, it is straightforward

to observe that fever or heartbeat are variables that can have similar values for one class and their neighbouring classes.

We have mentioned two main issues about ordinal classification: error characterization and data ordering exploitation. In addition, and starting from this illness classification example, some data characteristics widely recognized as challenging issues in machine learning can be pointed out. Firstly, class imbalance problem is also present in ordinal classification problems [17, 19, 40]. In the example, the number of patterns of class *risk* can be relatively small when compared to the number of patterns of *absence* class. Secondly, since class labels often represent ranks or degrees, class overlap can occur between all or a subset of classes. In the example, classes *risk* and *severe* will be probably highly overlapped. Specially for latent variable models, which are by far the most extended methods [35, 40], the presence of high dimensionality and nonlinearly separable data can make the mapping function ϕ result into complex models that impose highly nonlinear transformations from the input space to the latent space. Imposing too rigid models in these projections can derive into problems for classifying patterns in the classes boundaries, specially in the presence of the before mentioned noise or class overlapping, since patterns placed on those boundaries can be projected to the interval belonging to the wrong class.

Considering the ahead issues, we can synthesize the following open challenges that constitute the objectives of this thesis:

STATE-OF-THE-ART IN ORDINAL REGRESSION. Compared with nominal classification, ordinal regression is a machine learning field much less studied and explored. However, there are several works and literature dealing with this ordinal regression, which makes necessary to perform a proper analysis of them. A taxonomy of ordinal regression methods and an effort to gather and compare the main metrics for their evaluation would help to contextualize the proposals in the field.

CLASS IMBALANCE. Considering the obvious multi-class feature and some of the ordinal classification problems nature, the ordinal regression datasets present a high imbalance degree (i.e. some classes have few patterns when compared to the rest of the classes). Imbalance problem can generally harm classifiers, which tend to ignore minority populated classes, presenting a trivial behaviour with respect to those classes.

DATA ORDERING EXPLOITATION. Several authors define ordinal classifiers, and their associated learning algorithms, as a) methods that optimize the classification task constrained to metrics that consider errors ordering and magnitude, and b) methods which exploit the a priori knowledge about ordered placement of patterns. However, the second aspect of ordinal classifiers is not usually included explicitly in the classifier formulation.

Last, it should be highlighted we consider necessary not the only the proposal of new methods but also the application of the developed models and algorithms to real

world problems[4]. Then, real world ordinal regression problems will be identified and tackled by using the different methods proposed.

1.5 OBJECTIVES

The present thesis addresses the aforementioned open challenges, although the two latter are the main ones. The first challenge is a necessary starting point and the second one is not exclusive for ordinal classification. All these challenges result in the following formal objectives considered for the thesis:

1. State of the art in ordinal regression objectives:
   a) *To propose an OR method taxonomy*. The first part in the study consist on a review of the state of the art in ordinal regression. Although this objective is implicit in any thesis, ordinal regression is a very recent field and, up to the authors knowledge, there are not surveys related to this topic. Thereafter, it is even more important to collect existing works and also to propose a taxonomy to organize existing methods in order to properly contribute to the state-of-the-art. That is the reason why we consider necessary to formalize this objective.
   b) *To review OR evaluation metrics*. As pointed out in the Introduction, ordinal regression needs specific performance metrics to consider not only the number of errors but also the magnitude of the errors. An effort has to be made in order to gather all different metric proposals in this field.
   c) *To select benchmark datasets*. Preliminary exploration of the state of the art suggest that there are no public specific datasets repositories for ordinal classification. The most used dataset repository in the literature is the *ordinal regression benchmark datasets* provided by Chu and Ghahramani [41]. However, the benchmark datasets provided by Chu and Ghahramani are not real ordinal classification datasets but regression problems which target variable has been discretized. We identify two problems regarding these datasets: first, the datasets are not real classification datasets, therefore issues such as class mislabel or class imbalance related to the problems nature are not present here; and second, using same width intervals for classes label generation produce an artificial class imbalance in the datasets.
2. Class imbalance can be divided into the following objectives:
   a) *To perform an analysis of the state of the art for nominal class imbalance*. Class imbalance problem has been widely attracting scientific attention in the last decade. Then it is mandatory to briefly explore this work of the community before searching for new proposals.
   b) *To optimize algorithms that tackle the nominal class imbalance as a multi-objective optimization problem*. For dealing with class imbalance, multi-objective

4 Indeed, the AYRNA research group, to which the doctorate belongs to, has a demonstrated experience in the application of machine learning techniques to real world problems, see Publications in http://www.uco.es/ayrna

evolutionary algorithms can be considered as proposed in the works of Fernández-Caballero et al. [21] and Gutiérrez et al. [22]. The results of these works present interesting classification performance for all the classes in multi-class problems but their computational cost is expensive. Then it raises the issue of exploring more efficient approaches, such as Extreme Learning Machine (ELM) [42], which is a non-iterative algorithm for learning neural networks.

c) *To explore new solutions considering ordinal class imbalance*. As previously stated, ordinal regression datasets commonly suffer from the problem of class imbalance. The two previous objectives are aimed at helping to develop a new method for dealing with skewed class distributions in ordinal regression problems.

3. Data ordering exploitation. In this work, latent variable models will be drawn special attention to develop new ordinal classifiers which are able to better exploit data ordering, so that the following objectives are formulated:

   a) *To check if data ordering exploitation improves classification performance in OR problems*. Although the main premise is that OR classification should perform better than nominal classification for ordinal problems, experimental results should demonstrate a better performance of OR methods with respect to standard nominal classifiers.

   b) *To design OR algorithms based on standard regression but avoiding any trivial assumption about the latent variable*. As presented in this chapter, some simplifications [25] of the OR problem suggest using standard regression to predict classes in the ordinal scale. However, how to choose an optimal value to represent each class is an open issue, being dependent, in general, of the problem considered. One of the objectives of this thesis is to extend this previous proposals but avoiding any trivial assumption about the class labels.

   c) *To develop latent variable modelling approaches only with restrictions in the labels set*. The strict definition of ordinal regression tighten the order restriction to the target variable space. Then some of the proposals will be developed only with this restriction.

   d) *To develop classifiers that exploit the input data ordering*. In spite of this strict definition (see discussion in Section 3.2 of Chapter 3), some authors such as Hühn and Hüllermeier [17] have extended the definition of ordinal regression by suggesting that the labels ordering can be present in the input space. Then, algorithms and models could be benefited of exploiting this a priori assumption and we will study it with specific proposals.

   e) *To develop methods that relax the data projection of threshold methods*. As the reader can check in Chapter 3, most of methods build projection models mapping patterns to the latent space aiming at maximizing inter-class distances and minimizing intra-class distance, so that patterns of the same class are closely projected in the latent space, and patterns of different classes are projected as much separate as possible. Relaxing these condi-

tions could improve generalization performance, and will present a novelty in latent variable modelling.

4. Application of OR methods to real problems. In this thesis some real world problems will be addressed under the umbrella of OR. The ultimate goal is to justify the research in the field of OR. Here there are two objectives associated to two real problems:

   a) *To develop sovereign credit rating classification methods using ordinal regression*. The problem of the sovereign rating has had an increasing importance since the beginning of the financial crisis. However the credit rating agencies role in the financial crisis, among other factors, has motivated the research in this type of problems. Evaluations of creditworthiness are commonly formulated in an ordinal scale. Nevertheless, the OR regression approach for modelling these problems is being scarcely used, and binary classification has been the most common method applied in the financial and credit fields [43].

   b) *To develop wind speed forecasting systems using ordinal regression*. Previous works consider the wind speed as a continuous target variable, estimating then the corresponding wind series of continuous values. However, the exact wind speed is not always needed by wind farms managers, and a general idea of the level of speed is, in the majority of cases, enough to set functional operations for the farm. Wind speed can be described in an ordinal scale, and it seems interesting to evaluate the performance of OR methods for this problem.

1.6 ROAD MAP

The dissertation is organised as follows (each chapter will include a brief summary of its contents and the list of publications associated to it):

- Chapter 2: COMPUTATIONAL INTELLIGENCE FOR CLASSIFICATION AND REGRESSION introduces computational intelligence techniques that are used or extended during the present thesis.
- Chapter 3: ORDINAL REGRESSION presents the ordinal regression survey, including a taxonomy proposal for methods and a collection of performance metrics for OR. This chapter covers objectives 1*a* and 1*b*.
- Chapter 4: NEW PROPOSALS FOR CLASS IMBALANCE PROBLEM presents a literature review for class imbalance, and establishes the new proposals for efficient classifiers training with imbalanced sets. This chapter covers objectives 2*a* and 2*b*.
- Chapter 5: NEW PROPOSALS FOR ORDINAL REGRESSION. In this chapter the first proposal links the work with class imbalance and OR (objective 2*c*). The second proposal deals with the latent variable modelling as a regression target (objective 3*b*) with exploitation of the label ordering (goal 3*c*) and input space ordering (objective 3*d*). In this chapter the projection of the input data to the

latent space is relaxed as mentioned in objective 3*e*. Also, several datasets are collected and used, so that it also covers objective 1*c*.

- Chapter 6: APPLICATION OF ORDINAL REGRESSION TO SOVEREIGN CREDIT RATING presents the application of the PCDOC method (proposed at Chapter 5) to the problem of sovereign credit rating. In this chapter not only a classification system is proposed, but also the projection of the regressor model in PCDOC is proposed for ranking visualization, which might be suitable to build a decision support system. This chapter covers objective 4*a*.
- Chapter 7: APPLICATION OF ORDINAL REGRESSION TO WIND SPEED FORECASTING presents the application of OR techniques to wind forecasting to help wind farms managers. This chapter covers objective 4*b*.
- Chapter 8: SUMMARY, CONCLUSIONS, AND FUTURE WORK ends up this thesis dissertation with a summary of the scientific contributions and with hints about possible future research lines.

Finally, the experimental results in chapters 5, 6 and 7 cover objective 3*a* because they show a general performance improvement obtained by OR methods with respect to their nominal counterparts.

2

COMPUTATIONAL INTELLIGENCE FOR CLASSIFICATION AND REGRESSION

Summary. This chapter presents some of the background knowledge for this thesis. Specifically, both neural networks and Support Vector Machines are used as the base of the proposals, and this chapter aims to briefly introduce them, defining their formulation and describing their learning process.

2.1 INTRODUCTION

As mentioned in the Introduction, machine learning groups methods that learn for data in order to perform, in general, a classification or regression task. The goal is to build models in such a way they generalize as better as possible over unseen data. That is, the model will predict an output variable value according to new unlabelled data.

Previous to the rise of machine learning and computational intelligence, these models were obtained through techniques such as optimization methods which aim to minimize an error function. The researcher typically decided which method was suitable for the data and then applied the optimization method. Nevertheless, data nature is complex, and it presents handicaps such as non-linearity or high dimensionality which hints the modelling process.

With the ahead motivation, there arose techniques such as artificial neural networks or the statistical learning theory in the 1960's [44]. However, the lack of powerful computational resources made those proposals lay exclusively on the theoretical plane. In the 1980's, the re-discovering of the backpropagation algorithm, as well as hardware improvements, caused the rebirth of neural networks and the theoretical plane was overcome. Later, in the 1990's, other related techniques arose, such as Support Vector Machines [44], also with theoretical and practical applications.

In this thesis, we will develop extensions to existing neural network models and include the SVM for regression as a component of ordinal regression models. Moreover, we will employ several ANN and SVM methods to compare our proposals in the experimental section. Thereafter, this chapter provides basic background related to these techniques.

2.2 ARTIFICIAL NEURAL NETWORKS

2.2.1 *Definition of ANNs*

Artificial neural networks (ANN), or simply neural networks, [16, 45] are a flexible modelling technique which is based on the emulation of biological nervous systems. Then, ANNs try to emulate how the human brain solves problems, and therefore their biological interpretation is similar to the activation of brain neurons [46].

ANNs combine large amount of processing elements which are highly interconnected, and its computing capacity is developed through adaptive learning procedures that properly fit the ANN model. The processing elements are typically called *nodes* or *neurons* and they are structured in *layers*. The disposition allows ANNs to be flexible and to be able of modelling complex systems.

The general form of an ANN model is a *black box* in the sense that interpretability of cause-effect relationship cannot be easily obtained, and where attributes or input variables belong, in general, to a high dimensional space[1]. Neurons are modelled by means of *processing units* or *model's nodes*. Each processing unit consists of a set of input connections, an activation function (which computes a value from all the input connections), the central processing core (that applies the activation function), and finally an output or transfer function (which transfers the activation value to other units).

Mathematically, a neuron's network function $f(\mathbf{x})$ is composed of other functions $g_j(\mathbf{x})$, which can be also defined as a composition of functions. This can be graphically represented as a network structure, with arrows depicting the dependencies between nodes [15]. The composition typically used is the nonlinear weighted sum, so $f(\mathbf{x}) = h\left(\sum_j \mathbf{w}_j g_j(\mathbf{x})\right)$, where h is a predefined nonlinear activation function. In this way, the elements that define a node of an ANN are:

- The *weighted connections* perform the role of the synaptic connections. Connection existence determines where it is possible that a node influences other node, while signs and weight values define the type (excitatory or inhibitory) and intensity of the influence.
- The *activation function* calculates the base value or total input arriving to the node (generally as a simple weighted sum of all inputs, i.e. the sum of the inputs multiplied by the weights or connections values). We call a_j to the resulting value.
- The *transfer function*, also known as output function, calculates the node output as a function of the neuron activation. It is typically represented as $h(\cdot)$, and different types of functions may be used, from simple threshold functions to complex nonlinear functions. The output value ϕ_j of a hidden node is:

$$\phi_j(\mathbf{x}) = h(a_j) \tag{2.1}$$

being $\mathbf{x}$ the vector of input variables describing a pattern.

1 Input variables are sometimes called attributes of features, however when working with kernel spaces, *attributes* is reserved to the input space, and the unknown kernel space is refereed as the *feature space*

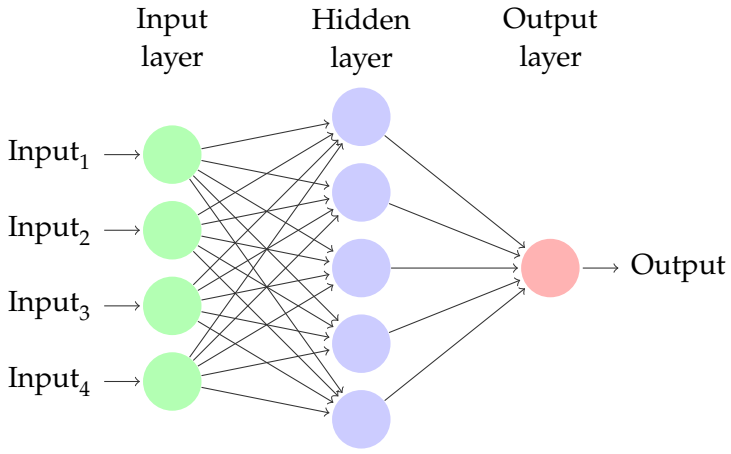


Figure 2.1: Example of a single layer feed-forward neural network model with three layers, four input variables, five hidden nodes and one output node[2].

The architecture of an ANN is the way its nodes are arranged. The nodes, according to their situation in the network can be of three types:

- *Input nodes* that receive network input signals (this is, the value of the model independent variables), and build the input layer.
- *Output nodes*, which send output signals (this is, the dependent or response variables), and make up the output layer.
- *Hidden nodes*, which are the rest of the nodes, and are grouped into one or more hidden layers.

There are several alternatives to organize the information transmission between nodes in the network, and the alternatives determine the nature of the ANN. Generally, ANNs can be grouped into:

- Feed-forward neural networks, in which information is propagated only from input nodes to output nodes, i.e. their graph is a directed acyclic graph.
- Recurrent neural networks, which propagates information from input to output nodes and also to other nodes, allowing the existence of cycles in the graph.

In this thesis, we will focus on single layer feed-forward (SLFF) neural networks, which are the simplest case with only three layers (input layer, output layer and one hidden layer). Figure 2.1 shows an example of this kind of ANNs. These can be formally defined as a linear regression model that considers a linear combination of nonlinear transformations of the input variables ($\phi_j(\mathbf{x}, \mathbf{w}_j)$), with the following expression:

$$y(\mathbf{x}, \theta) = \beta_0 + \sum_{j=1}^{M} \beta_j \phi_j(\mathbf{x}, \mathbf{w}_j) \tag{2.2}$$

where M is the number of nonlinear transformation, $\theta = \{\boldsymbol{\beta}, \mathbf{w}_1, \ldots, \mathbf{w}_M\}$ is the parameter set associated to the model, $\boldsymbol{\beta} = \{\beta_0, ..., \beta_M\}$ are the coefficient values associated to the lineal part of the model, $\phi_j(\mathbf{x}, \mathbf{w}_j)$ represent the *basis functions* and

$\mathbf{x} = \{x_1, ..., x_K\}$ is the vector containing the input or independent variables. This type of models are known as linear basis function models [9]. The polynomial regression is an example of these models in which there is only one input variable and each basis function is a power of this variable, $\phi_j(\mathbf{x}) = x^j$. An extension of these type of regression models are the spline functions regression methods, in which the input space is divided into different regions and each region is approximated with a different polynomial [47]. There are numerous options for selecting basis functions typology, for instance, Gaussian functions that result in radial basis functions (RBF) [48, 49] neural networks, sigmoidal units which produce the MultiLayer Perceptron (MLP) [50], or product unit functions [51].

2.2.2 *Taxonomy of neural networks according to the basis function*

In general terms, we can consider two classes of activation functions for neurons or nodes: additive and multiplicative. These types of functions result in two types of ANNs, whose peculiarities are discussed below:

- The *additive model* is the most used class. The output function of this neurons is

$$\phi_j(\mathbf{x}, \mathbf{w}_j) = h(w_{j0} + w_{j1}x_1 + w_{j2}x_2 + ... + w_{jK}x_K),$$

  where $\mathbf{w}_j = \{w_{j0}, w_{j1}, ..., w_{jK}\}$ represents the value of the coefficients associated with the node, $h(\cdot)$ the transfer function and w_{j0} the node activation threshold or bias. There are several kinds of additive nodes, for instance threshold neurons or perceptron [46] (which employ a step function), sigmoidal units (which consider functions such as logistic sigmoid, hyperbolic tangent or arctangent functions) and the lineal nodes (where the transfer function is the identity function).

- The *multiplicative model* is a more recent model which aims to model problems in which there is an interaction between variables and decision regions that cannot be separated by hyper-planes [52]. The more general approach here are the product units (PUs):

$$\phi_j(\mathbf{x}, \mathbf{w}_j) = x_1^{w_{j1}} \cdot x_2^{w_{j2}} \cdot ... \cdot x_K^{w_{jK}}$$

  being $\mathbf{w}_j = \{w_{j1}, w_{j2}, ..., w_{jK}\}$ because the bias parameter is nonsense sense here. Since the weights $w_{j1}, w_{j2}...w_{jK}$ are real numbers, PUs generalize other classes of multiplicative neurons.

On the other hand, considering the basis functions, we can distinguish between:

- *Kernel functions*, which are local functions such as RBFs. Those functions have better capacity for approximating isolated data, but they can perform worse in the global space or when the number of input variables is high.

- *Projection functions*, that are global functions, such as the sigmoidal or product unit. They tend to have more difficulties to deal with isolated data but their performance is in general better when the number of input dimensions grows.

Following sections summarize the most extended models for single layer feedforward neural networks.

2.2.3 *Sigmoidal unit neural networks*

A neural network with sigmoidal units or, more commonly called, a MultiLayer Perceptron (MLP), is that formed by sigmoidal units in the hidden layer. These nodes present an additive projection model with the following output function:

$$\phi_j(\mathbf{x}, \mathbf{w}_j) = \frac{1}{1 + e^{-(w_{j0} + w_{j1} \cdot x_1 + w_{j2} \cdot x_2 + \ldots + w_{jK} \cdot x_K)}} = \frac{1}{1 + e^{-(w_{j0} + \sum_{i=1}^{K} w_{ji} \cdot x_i)}}$$

The MLP networks have a important property: the family of real functions that represent those networks can approximate any function with enough precision if the proper number of hidden nodes is selected. This property provides a solid theoretic basis for the study, development and application of these networks [50, 53, 54].

2.2.4 *Product unit neural networks*

The product unit neural networks (PUNNs) were introduced by Durbin and Rumelhart in 1989 [51], and are those consisting of product unit nodes in the hidden layer. These units follow a multiplicative projection model with the output function:

$$\phi_j(\mathbf{x}, \mathbf{w}_j) = x_1^{w_{j1}} \cdot x_2^{w_{j2}} \cdot \ldots \cdot x_K^{w_{jK}} = \prod_{i=1}^{K} x_i^{w_{ji}}$$

where $\mathbf{w}_j = \{w_{j1}, w_{j2}, \ldots, w_{jK}\}$ because these ANN models do not have bias in the input layer.

Finally, as a consequence of the Stone–Weierstrass theorem[3], it is direct to prove that PUNNs are universal approximators (observe that polynomial functions in several variables are a subset of product unit models) [55, 56].

2.2.5 *Radial basis function neural networks*

A radial basis function (RBF) is a real-valued function whose value depends only on the distance from the origin, or alternatively on the distance from some other point called *center*. Then, RBF neural networks are those having RBF nodes in the hidden layer. Each one of the RBF nodes makes an independent local approximation of the input space, typically by a Gaussian function. The lineal output layer joins the effect of all the nodes by adding each obtained value in the hidden layer. The key idea is that each node is placed in a region in the input space (i.e. with center or mean in this region) and a specific radious or width. The learning process for RBFs consists on moving the hidden layer nodes through the input space varying the center and widths, in such a way the network is better adjusted to the training data.

The activation function is equivalent to the Euclidean distance function (where the RBF center is the vector $\mathbf{w}_j$), and the transfer function is generally a Gaussian function. Thereafter, the transfer function is:

$$\phi_j(\mathbf{x}, \mathbf{w}_j) = e^{-\frac{1}{2}\left(\frac{d(\mathbf{x}, \mathbf{w}_j)}{r_j}\right)^2},$$

3 The Weierstrass approximation theorem states that every continuous function defined on a closed interval $[a, b]$ can be uniformly approximated as closely as desired by a polynomial function.

where

$$d(\mathbf{x}, \mathbf{w}_j) = \| \mathbf{x} - \mathbf{w}_j \| = \sqrt{\sum_{i=1}^{K} (x_i - w_{ji})^2},$$

$\mathbf{w}_j = \{w_{j1}, w_{j2}, ..., w_{jK}\}$ is the center of the RBF and r_j is its radius or width.

RBF networks have also been proved to be universal approximators [57]. Compared with MLP, RBF neural networks have the advantage of having local elements in the model, and, as a consequence only some neurons are activated for a specific pattern. This facilitates the training process, and both local optima and error surface complexity are reduced because weight interactions are minimized. For concluding, MLP training generally consists on one single phase, while RBF networks typically needs two phases: first, basis functions are approximated by non supervised learning, and then, output weights are fitted by supervised learning [9].

2.2.6 *Functional model*

As mentioned, the ANN models used in this thesis are single layer feedforward networks. Those models consist of an input layer that receives the problem independent variables, a hidden layer with different types of nodes available, and a linear output layer with one or several output nodes.

2.2.6.1 *Functional model for regression*

An ANN used for regression tasks will have one output node (see example in Figure 2.2). In this way, if we represent the set of coefficients associated to the ANN as $\theta = \{\boldsymbol{\beta}, \mathbf{W}\}$, the functional model associated to the output node is:

$$f(\mathbf{x}, \theta) = \beta_0 + \sum_{j=1}^{M} \beta j \cdot \phi_j(\mathbf{x}, \mathbf{w}_j), \tag{2.3}$$

where $\boldsymbol{\beta} = \{\beta_0, \beta_1, \ldots, \beta_M\}$ is the set of weights from hidden layer nodes to the output node, β_0 is the bias coefficient (or network bias), $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M\}$ the set of coefficients associated to the hidden layer connections to the input layer, $\mathbf{w}_j = \{w_{j0}, w_{j1}, ..., w_{jK}\}$ the values of the connections of node j of hidden layer and $\phi_j(\mathbf{x}, \mathbf{w}_j)$ the output of node j of the hidden layer. In this way, any of the previously presented basis functions $\phi_j(\mathbf{x}, \mathbf{w}_j)$ could be used (considering that PU units do not include the bias). For RBFs, a common solution is to consider the centre of a node as $\mathbf{c}_j = \{w_{j1}, ..., w_{jK}\}$ and the radius as $r_j = w_{j0}$.

2.2.6.2 *Functional model for classification*

In a supervised classification problem, the purpose is to predict the class a pattern belongs to based on a training procedure with input numerical variables which are labelled according to its class. The measured (random observations) variables x_k, $k = 1, 2, ..., K$ (K is the number of input dimensions) are grouped as an unique pattern which should be classified in one of the Q classes based on these variables.

The, the training set is described as $\mathbf{D} = \{(\mathbf{x}_i, \mathbf{t}_i); i = 1, 2, ..., N\}$, where $\mathbf{x}_i = (x_{i1}, ..., x_{iK})$ is the input vector with input variables taking values in $\mathbf{x} \subset \mathbb{R}^K$ and

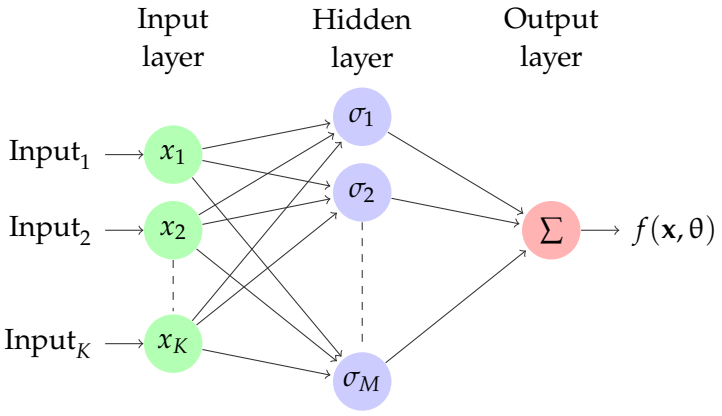


Figure 2.2: ANN model with K input neurons, M nodes in the hidden layer and one output node (typically used for regression tasks).

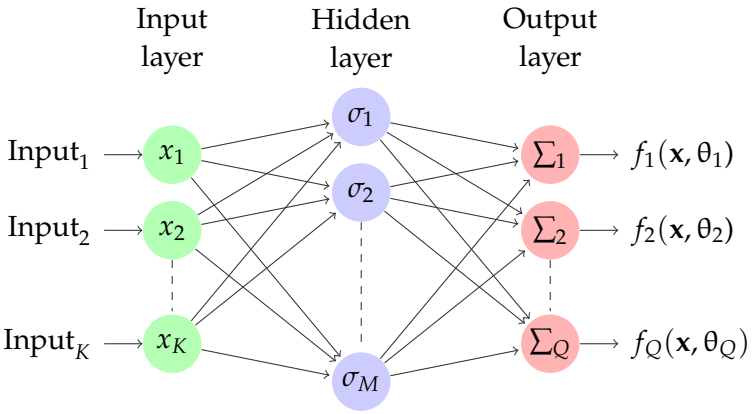


Figure 2.3: ANN model with K input neurons, M nodes in the hidden layer and Q output nodes (typically used for classification tasks, where Q is the number of classes).

$\mathbf{t}_i$ represents the class of the i-th pattern. When applying ANNs for nominal classification, the most usual approach is to consider a 1-of-Q coding scheme [9], i.e. $\mathbf{t}_i = \{t_{i1}, \ldots, t_{iQ}\}$, $t_{iq} = 1$ if $\mathbf{x}_i$ corresponds to an example belonging to class C_q, and $t_{iq} = 0$ (or $t_{iq} = -1$), otherwise. Then, an ANN for classification will have Q or $Q-1$ output nodes (see example in Figure 2.3). We can represent the coefficient associated to an ANN with $\theta = \{\theta_1, ..., \theta_Q\}$, then the functional model associated to each output node is:

$$f_q(\mathbf{x}, \theta_q) = \beta_{q0} + \sum_{j=1}^{M} \beta_{qj} \cdot \phi_j(\mathbf{x}, \mathbf{w}_j), \tag{2.4}$$

being $1 \leq q \leq Q$, $\theta_q = \{\boldsymbol{\beta}_q, \mathbf{W}\}$ the set of coefficients of the q output node, $\boldsymbol{\beta}_q = \{\beta_{q0}, \beta_{q1}, ..., \beta_{qM}\}$ the weights of connections between hidden and output layer for this node, $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M\}$ the set of coefficients associated to the hidden layer, $\mathbf{w}_j = \{w_{j0}, w_{j1}, ..., w_{jK}\}$ the values of the connections of node j of hidden layer and $\phi_j(\mathbf{x}, \mathbf{w}_j)$ the output of node j of the hidden layer.

The purpose of the training process is to find an estimation of the ANN coefficients $\hat{\theta}$ which allows the ANN to better classify unseen patterns. This is equivalent to obtaining a decision function $C : \{\boldsymbol{\Omega}, \theta\} \rightarrow \{1, 2, ..., Q\}$ for classifying individuals. In other words, the $\boldsymbol{\Omega}$ space provides a partition $\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_Q$, where $\mathbf{D}_q$ represents

the q-th class, $1 \leq q \leq Q$, and the patterns belonging to $\mathbf{D}_q$ will be assigned to class q. An incorrect classification is done if C assigns a pattern to class j when its actual class is q with $q \neq j$.

In order to value the performance of an ANN for classification (or other type of classifiers), typically the Correct Classification Rate (CCR) is used:

$$CCR(\hat{\theta}) = \frac{\sum_{i=1}^{N} I(C(\mathbf{x}_i, \hat{\theta}) = y_i)}{N} \tag{2.5}$$

where $I(g)$ is a function that returns 1 if g is true and 0 otherwise, and $C(\mathbf{x}_i, \hat{\theta})$ is the class that the network with coefficients $\hat{\theta}$ assigns to the pattern i. A good classifier maximizes the $CCR(\hat{\theta})$ ratio or minimizes the corresponding error, this is, $(1 - CCR(\hat{\theta}))$.

2.3 SUPPORT VECTOR MACHINES

The Support Vector Machines (SVM) [58] are perhaps the most common kernel learning method for statistical pattern recognition. This technique (SVM and its variants and extensions) has been studied extensively and applied to several pattern classification and approximation problems.

2.3.1 *Support vector machines for binary classification*

This section cover the basic ideas of SVM, that initially were formulated for binary classification. The SVM can be extended to multi-class problems by means of different mechanisms, for instance binary decompositions such as the *OneVsOne* scheme. This topic is briefly explained in sections 3.3.2 and 3.3.2.1 in Chapter 3.

2.3.1.1 *Hard-margin support vector machines*

The basic idea behind SVMs is to separate the two different classes through the optimal separating hyperplane which is specified by its normal vector $\mathbf{w}$ and the bias term b, so if the training data is linearly separable, we could determine the separating hyperplane as:

$$\mathbf{w} \cdot \mathbf{x}^T + b = 0, \tag{2.6}$$

what yields the corresponding decision function:

$$\text{if } \mathbf{w}^T \cdot \mathbf{x} + b \begin{cases} > 0, & y_i = 1, \\ < 0, & y_i = -1, \end{cases} \tag{2.7}$$

where the labels for binary classification are in the set $\{-1, 1\}$.

Figure 2.4 shows different separating hyperplanes (H_1, H_2, H_3) for addressing the classification problem. As seen, the green hyperplane (H_3) performs a trivial classification, unlike the red one (H_2) and the blue one (H_1). But when comparing the red and the blue ones, both perform a perfect classification. However, the red hyperplane (H_2) would be preferred because it maximizes the margin between classes. This property makes it more robust when classifying future patterns.

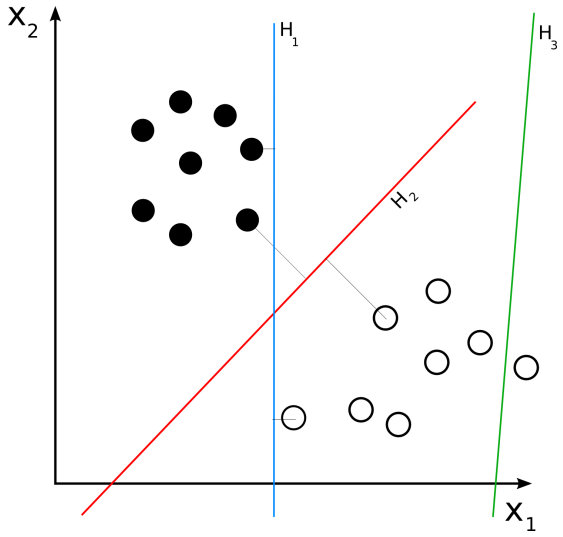


Figure 2.4: Different separating hyperplanes computed for the classification.

If we take into account that no training data satisfy $\mathbf{w}^T \cdot \mathbf{x} + b = 0$ (as they are linearly separable), we could consider instead the following inequalities for the sake of separability:

$$\text{if } \mathbf{w}^T \cdot \mathbf{x} + b \begin{cases} \geq 1, & y_i = 1, \\ \leq -1, & y_i = -1. \end{cases} \tag{2.8}$$

Then, Eq. 2.8 would be equivalent to:

$$y_i(\mathbf{w}^T \cdot \mathbf{x} + b) \geq 1 \quad \text{for } i = \{1, \ldots, N\}. \tag{2.9}$$

The distance between the separating hyperplane and the training data sample nearest to the hyperplane is called the *margin*. Now, consider determining the optimal separating hyperplane. The distance from a training data sample $\mathbf{x}$ to the separating hyperplane is given by $|D(\mathbf{x})|/||\mathbf{w}||$. Because the vector $\mathbf{w}$ is orthogonal to the separating hyperplane, the line that goes through $\mathbf{x}$ and that is orthogonal to the hyperplane is given by $\frac{a\mathbf{w}}{||\mathbf{w}||} + \mathbf{x}$, where $|a|$ is the distance from $\mathbf{x}$ to the hyperplane.

Finally, to find the optimal separating hyperplane we need to find $\mathbf{w}$ with the minimum distance norm that satisfies Eq. 2.9. Therefore, the optimal separating hyperplane can be obtained by solving the following minimization problem for $\mathbf{w}$ and b:

$$\text{minimize: } Q(\mathbf{w}, b) = \frac{1}{2}||\mathbf{w}||^2 \tag{2.10}$$

$$\text{s.t.: } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad \text{for } i = \{1, \ldots, N\}. \tag{2.11}$$

In Eq. 2.10, $||\mathbf{w}||$ is the Euclidean norm (as it is the one used in this case), and the square in the equation is added for avoiding an optimization problem with a square root. The assumption of linear separability means that there exist some $\mathbf{w}$ and b that satisfy Eq. 2.11 (this set of possible solutions for the problem are called *feasible solutions*). By using the quadratic objective function with the inequality constraints, even if the solutions are not unique, the value of the objective function will be unique. Thus, non uniqueness is not a problem for support vector machines, and this is

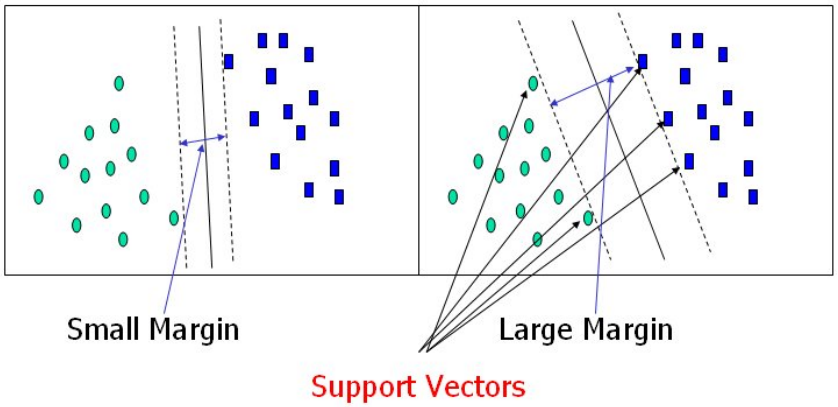


Figure 2.5: Graphical representations of the concepts *large margin* and *support vectors*.

precisely one of the advantages of support vector machines over neural networks, which have numerous local minima.

As can be seen in Figure 2.5, we are seeking for the optimal separating hyperplane, that is, the one which performs the larger separation for the training data. This figure shows two decision functions that satisfy 2.9. Thus, there are an infinite number of decision functions for that. The generalization ability will depend on the location of this separating hyperplane. If we assume that there are no outliers in the data and that the unknown test patterns obey the same distribution as that of the training data, then it is well-known that the generalization ability is maximized by using this optimal separating hyperplane.

The optimization constrained problem stated in Eq 2.10 and 2.11 could be converted into the unconstrained problem:

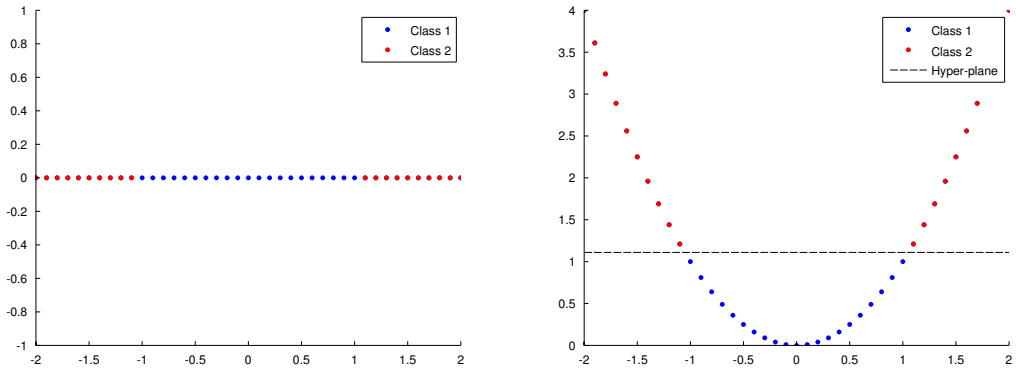
$$Q(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \{y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1\}, \tag{2.12}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)^T$ and α_i are the non-negative Lagrange multipliers.

2.3.1.2 *The kernel trick*

Before continuing, we need to introduce the concept of *kernel trick*, which is essential for SVM techniques (and other kernel methods). Basically, the *kernel trick* allows the computation of dot products in high-dimensional feature spaces, using simple functions defined on pairs of input patterns. This trick allows the formulation of nonlinear variants of any algorithm that can be cast in terms of dot products. The basic motivation for working in high dimensional spaces is that it can help to linearly separate data which is not linearly separable in the input space. An intuitive example of how increasing the number of dimensions can allow linear separability of the data is done in Figure 2.6.

The kernel concept was introduced into the field of pattern recognition by Aizerman et al. [59] in the context of the method of potential functions. Although neglected for many years, it was re-introduced into machine learning in the context of large margin classifiers by Boser et al. [60], giving rise to the technique of support



(a) Input space.

(b) Feature space.

Figure 2.6: Example of how increasing the number of dimensions can allow linear separability of the data. The example in the left side is not linearly separable. However, by adding the square of the input variable as an extra dimension, the data can be linearly separable.

vector machines. Since then, there has been considerable interest in this topic, both in terms of theory and applications. One of the most significant developments has been the extension of kernels to handle symbolic objects, thereby greatly expanding the range of problems that can be addressed. The general idea is that, if we have an algorithm formulated in such a way that the input vectors $\mathbf{x}_i$ appear only in the form of dot products between them, we can replace that dot product with some other choice of kernel. For instance, the technique of kernel substitution can be applied to principal component analysis (PCA) in order to develop a nonlinear variant of PCA [61]. Other examples of kernel substitution include nearest-neighbor classifiers and the kernel Fisher discriminant [62, 63].

When referring to a pattern classification algorithm as support vector machines or discriminant analysis, the optimal separating or projection hyperplane is supposed to maximize the accuracy. However, if the training data are not linearly separable, the obtained classifier may not present a high accuracy, even when the hyperplane is optimally determined.

Because of that, to enhance the linear separability, the original input space is mapped into a higher dimensional dot-product space (also known as feature space) by using a mapping function ϕ. Then, the linear decision function in the input space is given by:

$$D(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b, \tag{2.13}$$

where $\mathbf{w}$ determines the decision hyperplane and b is a bias term.

Figure 2.7 shows other toy example of a binary classification problem mapped into feature space. In this example, we assume that the true decision boundary is an ellipse in the input space. The task of the learning process is to estimate this boundary based on empirical data consisting on training points in both classes (crosses and circles, respectively). When mapped into feature space via the nonlinear mapping function ϕ, the ellipse becomes a hyperplane. This is due to the fact that ellipses can be written as linear equations in the entries of (z_1, z_2, z_3). Therefore, in the feature

space, the problem reduces to that of estimating a hyperplane from the mapped data points.

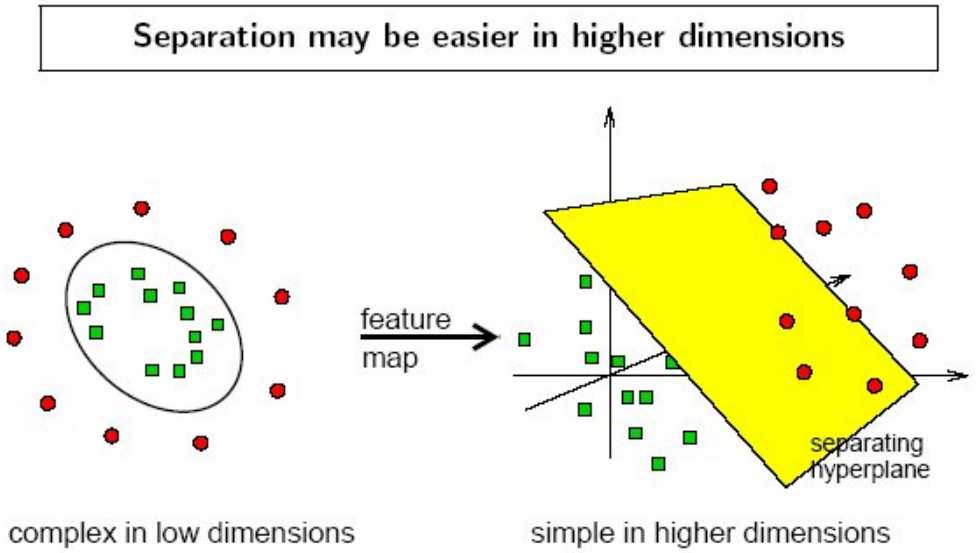


Figure 2.7: Toy example of a binary classification problem mapped into feature space.

The development of kernel-based algorithms has resulted from a combination of machine learning theory, optimization algorithms from operations research and kernel techniques from mathematical analysis. These three ideas have spread far beyond the original support vector machine algorithm: virtually every learning algorithm has been redesigned to exploit the power of kernel methodology.

But the main idea and advantage of the kernel methods is that without knowing the nonlinear feature mapping or the mapped feature space explicitly, we can work on the feature space through kernel functions, as long as the problem formulation depends only on the inner products between data points.

2.3.1.3 *Nonlinear support vector machine*

As said before, when data is not linearly separable, the *kernel trick* can be applied (using a mapping function ϕ) so that the previous definitions should be redefined. By doing this, the optimal separating hyperplane can be given as:

$$\mathbf{w} \cdot \phi(\mathbf{x}) + b = 0, \tag{2.14}$$

what yields the corresponding decision function:

$$f(\mathbf{x}) = \hat{y} = \operatorname{sgn}\left(\langle \mathbf{w} \cdot \phi(\mathbf{x}) \rangle + b\right),$$

where $\hat{y} = +1$ if $\mathbf{x}$ belongs to the corresponding class and $\hat{y} = -1$ otherwise. And, as done before, we should formulate the optimization problem and apply Lagrange multipliers.

Figure 2.8 shows the decision function after applying the kernel trick and the selected support vectors. This technique can be seen then as a linear parametric model re-cast into an equivalent dual representation in which the predictions are based on

a linear combination of a kernel function evaluated at the training data points. The parameters of the kernel model would be typically given by the solution of a convex optimization problem, so there is a single, global optimum.

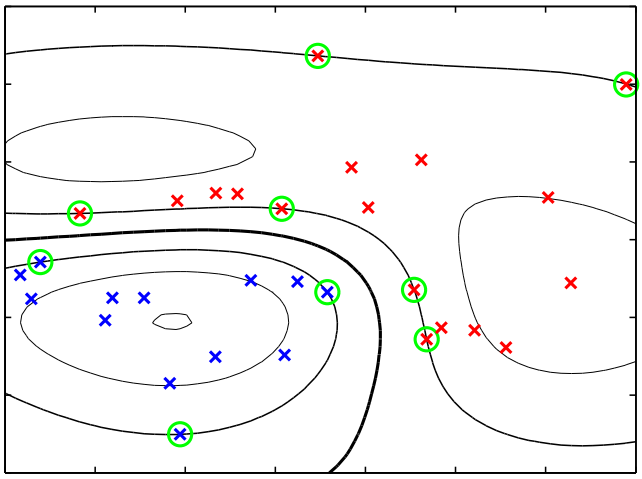


Figure 2.8: Graphical representation of the separation bounds after applying the kernel trick and the selected support vectors (source Bishop [9]).

SVMs can also be thought as generalized perceptrons with a kernel that computes the inner product on transformed input vectors $\phi(\mathbf{x})$, where $\phi(\mathbf{x})$ denote the feature vector $\mathbf{x}$ in a high dimensional reproducing kernel Hilbert space (RKHS) related to $\mathbf{x}$ by a specific transformation [58]. All computations are done using the reproducing kernel function only, which is defined as:

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x}) \cdot \phi(\mathbf{x}'),$$

where $\cdot$ denotes the inner product in the RKHS.

2.3.1.4 *Soft-margin support vector machines*

Up to this point, we have formulated the hard-margin support vector machine. Beyond specifying nonlinear discriminants by kernels, another generalization has been proposed which replaces hard margins by soft margins. This allows to handle noise and pre-labeling errors, which often occur in practice. If we do not handle this kind of problems, the hard-margin support vector machine can be unsolvable or it can lead to overfitting (when using the nonlinear version). Because of that, the so called slack-variables ξ_i are used to relax the hard-margin constraint [58]. Therefore, to allow inseparability, the nonnegative slack variables are introduced into Eq. 2.9, so that:

$$y_i(\mathbf{w}^T \cdot \mathbf{x} + b) \geq 1 - \xi_i \quad \text{for } i = \{1, \ldots, N\}. \tag{2.15}$$

By using the slack variables ξ_i, feasible solutions will always exist. For the pattern $\mathbf{x}_i$, if $0 < \xi_i < 1$ (as the case ξ_1 in Figure 2.9), the data is very close to the decision boundary but are still well-classified. But if $\xi_i \geq 1$ (as the case ξ_2 in Figure 2.9) the data are misclassified by the optimal hyperplane.

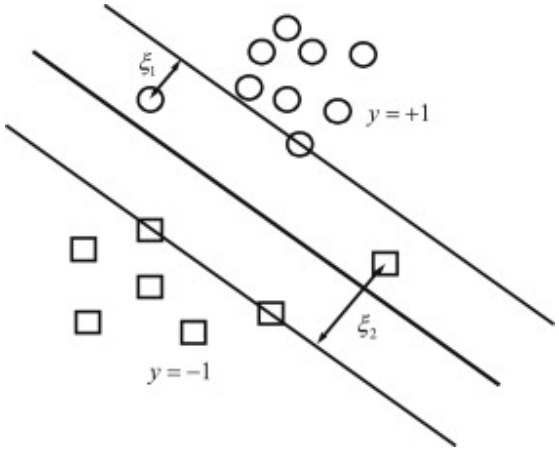


Figure 2.9: Graphical representation of the slack variables which allows the soft-margin classification.

So, to obtain the optimal hyperplane in which the number of training data that lie into the margin or into the wrong side of the hyperplane is minimum, we need to minimize:

$$Q(\mathbf{w}) = \sum_{i=1}^{N} \boldsymbol{\theta}(\xi_i),$$

where

$$\boldsymbol{\theta}(\xi_i) = \begin{cases} 1 & \text{for } \xi_i > 0 \\ 0 & \text{for } \xi_i = 0 \end{cases}.$$

But this is a combinatorial optimization problem and difficult to solve. Instead, we consider the following minimization problem:

$$\min_{\mathbf{w}\in\mathbb{R}^K} L(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i, \tag{2.16}$$

subject to:

$$y_i \cdot (\mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}) + b) \geq 1 - \xi_i,\ \xi_i \geq 0,\ \forall i = \{1, \cdots, N\}, \tag{2.17}$$

where y_i is the class of the input pattern $\mathbf{x}_i$, and constant C is a hyper-parameter that should be optimally selected (typically by a grid search procedure). The corresponding dual problem implies the use of Lagrange multipliers (α_i). More details can be found in [9].

The final classification rule can be expressed as follows:

$$\hat{y}(\mathbf{x}) = \text{sgn}\left(\sum_{i=1}^{N} \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i) + b\right), \tag{2.18}$$

where α_i represents the solution to the dual problem. Sparsity of the model is achieved because only a subset points $\mathbf{x}_j \in SV$ will result in nonzero Lagrange multipliers ($\alpha_j \neq 0$), where SV is the set of support vectors, so the model can be rewritten as:

$$\hat{y}(\mathbf{x}) = \text{sgn}\left(\sum_{\mathbf{x}_j\in SV} \alpha_j y_j k(\mathbf{x}, \mathbf{x}_j) + b\right), \tag{2.19}$$

The example of Figure 2.8 also provides a geometrical insight into the origin of sparsity in the SVM. The maximum margin hyperplane is defined by the location of the support vectors. Other data points can be moved around freely (so long as they remain outside the margin region) without changing the decision boundary, and so the solution will be independent of such data points [9].

2.3.2 *Support vector machines for regression*

Support vector machines have been extended to regression problems resulting to Support Vector Regression (SVR). This algorithm preserves the property of sparseness. First we define the error function to minimize. In the linear regression case, the following regularized error function is commonly used:

$$\frac{1}{2}\sum_{i=1}^{N}(\hat{y}(\mathbf{x}_i)-y_i)^2+\frac{\lambda}{2}\|\mathbf{w}\|^2. \tag{2.20}$$

In the case of SVR, the quadratic error function is replaced by an *ϵ-insensitive error function* proposed by Vapnik [64] in order to obtain a sparse solution. This function gives zero error if the absolute difference between the prediction $\hat{y}(\mathbf{x})$ and the target y is less than ϵ, where $\epsilon > 0$. A simple example of this error function with a linear cost associated with errors outside the insensitive region is (see Figure 2.10):

$$E_\epsilon(\hat{y}(\mathbf{x})-y)=\begin{cases}0, & \text{if } |\hat{y}(\mathbf{x})-y|<\epsilon\\ |\hat{y}(\mathbf{x})-y|-\epsilon, & \text{otherwise.}\end{cases} \tag{2.21}$$

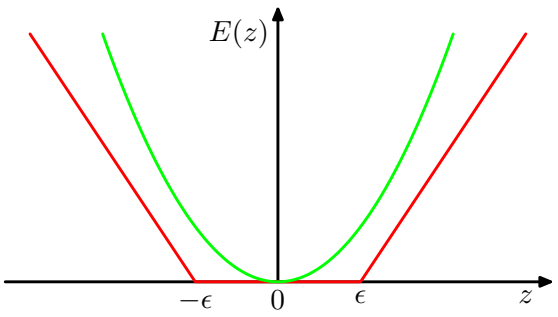


Figure 2.10: Plot of an ϵ-insensitive error function (red colour) in which the error increases linearly with distance beyond the insensitive region $[-\epsilon, +\epsilon]$. For comparison, it is also shown the quadratic error function (green colour) (source [9]).

Then, we minimize a regularized error function given by:

$$C\sum_{i=1}^{N}E_\epsilon(\hat{y}(\mathbf{x}_i)-y_i)-\frac{1}{2}\|\mathbf{w}\|^2. \tag{2.22}$$

where $\hat{y}(\mathbf{x}_i)$ is given by Eq. 2.13. By convention, the (inverse) regularization parameter C appears in front of the error term. We can reformulate the optimization problem by introducing slack variables but, in this case, we need two slack variables for each data point $\mathbf{x}_i$: $\xi_i \geq 0$ and $\xi_i^* \geq 0$, where $\xi_i > 0$ corresponds to a point for which target

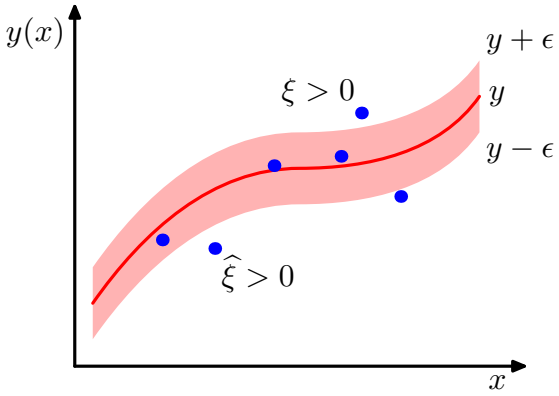


Figure 2.11: SVM regression model together with the ϵ-insensitive 'tube' (source [9]). ξ_i and ξ_i^* (ξ and $\hat{\xi}$ in the figure) are slack variables examples. Points above the ϵ-tube have $\xi_i > 0$ and $\xi_i^* = 0$, points below the ϵ-tube have $\xi_i = 0$ and $\xi_i^* > 0$. Points inside the ϵ-tube have $\xi_i = \xi_i^* = 0$.

value $y_i > \hat{y}(\mathbf{x}_i) + \epsilon$, and $\xi_i^* > 0$ corresponds to a point for which $y_i < \hat{y}(\mathbf{x}_i) - \epsilon$. An illustration of the SVM model for regression is shown in Figure 2.11.

A target point lies inside the ϵ-tube if $\hat{y}_i(\mathbf{x}_i) - \epsilon \leq y_i \leq y_i + \epsilon$. After introducing the slack variables, we allow points to lie outside the tube provided that the slack variables are non-zero. The corresponding conditions are:

$$y_i \leq \hat{y}_i(\mathbf{x}_i) + \epsilon + \xi_i,$$
$$y_i \geq \hat{y}_i(\mathbf{x}_i) - \epsilon - \xi_i^*.$$

Now the error function for support vector regression can be written as:

$$C\sum_{i=1}^{N}(\xi_i + \xi_i^*) + \frac{1}{2}\|\mathbf{w}\|^2,$$

which must be minimized subject to the constraints $\xi_i \geq 0$ and $\xi_i^* \geq 0$. The corresponding dual problem for minimizing this function implies using two kinds of Lagrange multipliers (α_i and α_i^*). More details can be found in [9].

Predictions of new inputs can be expressed in terms of the kernel function:

$$y(\mathbf{x}) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)\phi(\mathbf{x}, \mathbf{x}_i) + b,$$

where α_i and α_i^* are the Lagrange multipliers.

Figure 2.12 shows the influence of the ϵ parameter in the ϵ-SVR model. This gives us a hint about the relevance of this parameter that should be mandatory optimized (typically by a grid search cross-validation procedure).

2.4 RESOURCES

There a numerous books and online resources that the reader can check in order to extend its knowledge about artificial intelligence, machine learning, computational intelligence, pattern recognition and other fields.

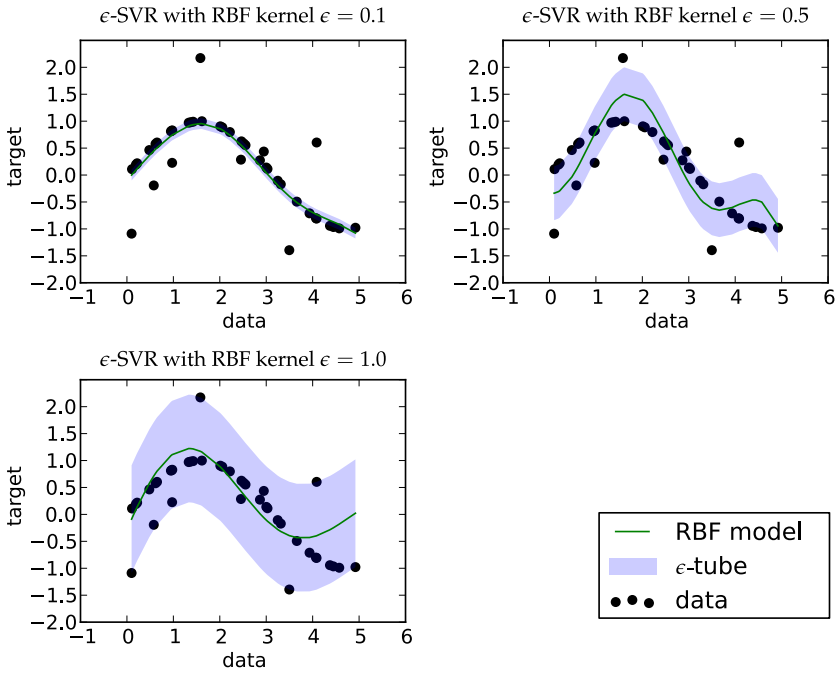


Figure 2.12: Example of the influence of the ϵ parameter in the SVR model.

2.4.1 *Books*

The following books are essential literature for machine learning and computational intelligence:

- Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer. 2007. ISBN 978-0-387-31073-2. Website with complementary resources http://research.microsoft.com/en-us/um/people/cmbishop/prml/.
- Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, 1st edition, December 2001. ISBN 0262194759.
- Agresti, A. Analysis of ordinal categorical data Wiley, 2010.

2.4.2 *On-line resources*

The following on-line resources are interesting for introducing many of the terms included in machine learning and computational intelligence:

- Machine learning course by Professor Andrew Ng (Stanford University) https://www.coursera.org/course/ml
- Neural Networks for Machine Learning by Professor Geoffrey Hinton (University of Toronto) https://www.coursera.org/course/neuralnets
- Artificial intelligence. (2013, May 13). In Wikipedia, The Free Encyclopedia. Retrieved 10:19, May 14, 2013, from http://en.wikipedia.org/w/index.php?title=Artificial_intelligence&oldid=554962155

- Machine learning. (2013, May 3). In Wikipedia, The Free Encyclopedia. Retrieved 10:22, May 14, 2013, from http://en.wikipedia.org/w/index.php?title=Machine_learning&oldid=553372918

2.4.3 *Free software*

The following free software integrates several machine learning and statistical methods:

- *Scikit-learn* (http://scikit-learn.org) [65] has been used for generating some of the graphics of this thesis, and presents a large collection of examples with special attention to the influence of hyper-parameters in the methods behaviour.
- *Weka 3: Data Mining Software* (http://www.cs.waikato.ac.nz/ml/weka/) [66] is a collection of machine learning algorithms for data mining tasks. In this thesis we have intensively used Weka for experimental comparisons as well as data pre-processing.

3

ORDINAL REGRESSION

Summary. This chapter presents a review of the state of the art of ordinal regression techniques, including a taxonomy proposal based on how the models are constructed to take the order into account. Up to the authors knowledge there are not similar reviews in this field.

Moreover, the chapter presents the mathematical notation generally used for the rest of the thesis, with the exception of some sections.

Associated publications. Some portions of this chapter appeared in [19, 40]:

- P.A. Gutiérrez, M. Pérez-Ortiz, F. Fernandez-Navarro, J. Sánchez-Monedero, and César Hervás-Martínez. An Experimental Study of Different Ordinal Regression Methods and Measures. In *7th International Conference on Hybrid Artificial Intelligence Systems*, pages 296–307, 2012. http://dx.doi.org/10.1007/978-3-642-28931-6_29.
- M. Cruz-Ramírez, C. Hervás-Martínez, J. Sánchez-Monedero, and P.A. Gutiérrez. A Preliminary Study of Ordinal Metrics to Guide a Multi-Objective Evolutionary Algorithm. In *Proceedings of the 11th International Conference on Intelligent Systems Design and Applications (ISDA 2011)*, pages 1176–1181, Cordoba, Spain, 2011. http://dx.doi.org/10.1109/ISDA.2011.6121818

Submitted publications. The review and taxonomy presented has been extended in a journal paper, which is under review process:

- P.A. Gutiérrez, M. Pérez-Ortiz, J. Sánchez-Monedero, F. Fernández-Navarro and C. Hervás-Martínez. Ordinal regression methods: survey and experimental study. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (Under Review).

3.1 INTRODUCTION

Learning to classify or to predict numerical values from prelabelled patterns is one of the central research topics in machine learning and data mining [67, 68]. However, less attention has been paid to ordinal regression (also called ordinal classification) problems, where the labels of the target variable exhibit a natural ordering.

For example, retrieving the weather prediction example of the introduction, the following ordinal scale $\{Very\ cold, Cold, Mild, Hot, Very\ hot\}$ represents an ordinal re-

gression problem since there is a natural order between classes: *Very cold* $\prec$ *Cold* $\prec$ *Mild* $\prec$ *Hot* $\prec$ *Very hot*. Then a sample vector associated with class label *Mild* has a higher label (or warmer temperature) than another from the *Cold* class, but *Hot* class is warmer than both. Thereafter, class labels are embedded with order information.

Ordinal regression problems present two issues to be addressed: misclassification costs are not the same for different errors (it is straightforward to think that predicting class *Hot* when the real class is *Cold* represents a more severe error than that associated with a *Very cold* prediction) and the ordering information can be used to construct more accurate models. Thus, specialized measures are needed for evaluating ordinal classifiers performance [18][19].

A further distinction is made by Anderson [69], which differentiates two major types of ordinal categorical variables, "grouped continuous variables" and "assessed ordered categorical variables". The first one is a discretized version of an underlying continuous variable, which could be observed itself, e.g. if age or income are only given in categories. The second one covers those variables where an expert or user provides his judgement on the grade of the ordered categorical variable. However, ordering is meaningful for both cases.

Ordinal regression problems are very common in many research areas, and they have been frequently considered as standard nominal problems which can lead to non-optimal solutions. Indeed, ordinal regression problems are between classification and regression, presenting some similarities and differences. Some of the fields where ordinal regression is found are medical research [4, 70–76], age estimation [77], brain computer interface [78], credit rating [6, 79, 80], econometric modelling [81], face recognition [82–84], wind speed prediction [5], facility layout design [85], social sciences [86], and more. All these works (which will be further analysed later in this chapter) are examples of application of specifically designed ordinal regression models, where the ordering consideration improves their performance with respect to their nominal counterparts. Additionally, a great research community is working in the Information Retrieval (IR) field motivated by the huge amount of data available [87]. Several works of ordinal regression have been motivated by or applied to IR. However, the two intrinsic properties of IR measures for ranking (query level and position based) cannot be sufficiently considered when using ordinal regression methods [87].

In statistics, ordinal data were firstly studied by using a link function able to model the underlying probability for generating ordinal labels [69]. The field of ordinal regression has evolved in the last decade, with noteworthy research progress made in supervised learning [88], from Support Vector Machines formulations [33, 37] to Gaussian processes [41] or discriminant learning [89], to name a few. However, to our knowledge, the methods have not yet been categorized in a general taxonomy, which is essential for further research and for identifying the developments made and the present state of existing methods. This chapter contributes a review of the state-of-the-art of ordinal regression and a taxonomy proposal to better organize the advances in this field.

Strongly related to ordinal regression, multicriteria decision analysis (MCDA) is an important field within operational research and is concerned with structuring and solving decision and planning problems involving multiple criteria. These methods are able to support decision makers facing such problems. Typically, there does

not exist a unique optimal solution, and decision maker's preferences are used to differentiate between solutions.

While both classification and sorting refer to the assignment of a set of alternatives into predefined groups, they differ with respect to the way that the groups are defined [90]: classification refers to the case where the groups are defined in a nominal way, and sorting refers to the case where the groups are defined starting from the most preferred alternatives and going on to the least preferred ones. A recent literature review [91] of the treatment of ordinal data in the field of MCDA includes the most significant advances, as well as connections with artificial intelligence. This thesis chapter focuses only on artificial intelligence approaches, so we refer to [91] for a detailed review of MCDA methods.

Recently in [17], ordinal meta-models were compared with respect to their nominal counterparts to check their ability to exploit ordinal information. The work concludes that such meta-methods do exploit ordinal information and may yield better performance. Another recent study [92] argues that ordinal classifiers may not have meaningful statistical advantage over corresponding non-ordinal counterparts, based on accuracy and Cohen's Kappa statistic [93]. Regarding both issues, experimental results in this thesis reveals that specifically designed ordinal regression methods can further improve the results with respect to meta-model approaches and with respect to nominal classifiers (see experimental results in Section 5.4.7 in Chapter 5, Section 6.3.4 in Chapter 6 and Section 7.3.5 in Chapter 7). There are statistical significant differences when using measures which take the order of the categories into account, which is the case of the Averaged Mean Absolute Error (AMAE). The aforementioned review paper submitted to a journal performs more extensive experiments including more methods and datasets, and therefore it enforces these preliminary conclusions.

3.2 NOTATION AND NATURE OF THE PROBLEM

3.2.1 *Problem definition*

The ordinal regression problem consists on predicting the label y of an input vector $\mathbf{x}$, where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^K$ and $y \in \mathcal{Y} = \{C_1, C_2, \ldots, C_Q\}$, i.e. $\mathbf{x}$ is in a K-dimensional input space and y is in a label space of Q different labels corresponding to the categories. The objective is to find a classification rule or function $f : \mathcal{X} \to \mathcal{Y}$ to predict the labels of new patterns, given a training set of N points, $\mathbf{D} = \{(\mathbf{x}_i, y_i), 1 \leq i \leq N\}$. All these considerations are common with standard nominal classification, but a natural label ordering is included for ordinal regression, $C_1 \prec C_2 \prec \ldots \prec C_Q$, where $\prec$ is an order relation given by the nature of the classification problem. Many ordinal regression measures and algorithms consider the rank of the label, i.e. the position of the label in the ordinal scale, which can be expressed by the function $\mathcal{O}(\cdot)$, in such a way that $\mathcal{O}(C_q) = q, 1 \leq q \leq Q$.

The difference between this setting and other related ones is now established. The assumption of an order between class labels makes that two different elements of $\mathcal{Y}$ can be always compared by using the relation $\prec$, which is not possible under the nominal classification setting. If compared to regression (where $y \in \mathbb{R}$), it is true that real values in $\mathbb{R}$ can be ordered by the standard $<$ operator, but labels in

ordinal regression ($y \in \mathcal{Y}$) do not carry metric information, so the category serves as a qualitative indication of the pattern rather than a quantitative one.

3.2.2 *Ordinal regression in the context of ranking and sorting*

Although ordinal regression has been paid attention recently, the amount of related research topics is worth to be mentioned. First, it is important to remark the differences between ordinal regression and other related ranking problems. There are three terms to be clarified:

- *Ranking* generally refers to those problems where the algorithm is given a set of ordered labels [94], with one label for each pattern, and the objective is to learn a rule that is able to rank patterns during the test phase by using this discrete set of labels. The induced ordering should be partial with respect to the patterns, in the sense that ties are allowed (two patterns with the same label). This rule should be able to obtain a good ranking, but not to classify patterns in the correct class. For example, if the labels predicted by a classifier are shifted one category (in the ordinal scale) with respect to the actual ones, the classifier will still be a perfect ranker.

- Another term, *sorting* [94] is referred to the problem where the algorithm is given a total order for the training dataset and the objective is to rank new sets during the test phase. As we can see, this is equivalent to a ranking problem where the size of the label set is equal to the number of training points, $Q = N$. Ties are not allowed for the prediction. Again, the interest is in learning a function that can give a total ordering of the patterns instead of a concrete label.

- The *multipartite ranking* problem (which is a generalization of the well know bipartite ranking one) deserves special attention. This kind of problem can be seen as an intermediate point between ranking and sorting. It is similar to ranking because training patterns are labelled with one of Q ordered ratings ($Q = 2$ for bipartite ranking), but here the goal is to learn from them a ranking function able to induce a total order in accordance with the given training ratings [95–98], which is similar to the sorting setting. The relationship between multipartite ranking and ordinal classification is discussed in [98]. An ordinal regression classifier can be used, of course, as a ranking function by interpreting the class labels as scores. However, this type of scoring will produce a large number of ties (which is not desirable for multipartite ranking). On the other hand, a multipartite ranking function $f(\cdot)$ can be turned into an ordinal classifier by deriving thresholds to define an interval for each class, but how to find the optimal thresholds is an open issue.

A more general term which is *learning to rank*, gathering different methods in which the goal is to automatically construct a ranking model from training data [87]. In this context, we refer now to the work of Lin and Li [31], which establishes different families of ranking model structures: *pointwise* or *itemwise ranking* (where the relevance of an input vector $\mathbf{x}$ is predicted by using either real-valued scores or ordinal-valued labels), *pairwise ranking* (where the relative order between two input vectors $\mathbf{x}$ and $\mathbf{x}'$

tries to be predicted, i.e. the local comparison nature of ranking, which can be easily cast to binary classification) and *listwise ranking* (where the algorithms try to order a finite set of patterns $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ by minimizing the inconsistency between the predicted permutation and the training permutation [87]). Both pairwise and listwise structures have serious problems of scalability with the size of the training dataset, and many works focus on trying to alleviate them [24, 99, 100]. However, in some cases, this can result in the loss of data information [31].

In summary, ordinal regression is a pointwise approach to classify data, where the labels exhibit a natural order. It is related to the problems of ranking, sorting and multipartite ranking, but, during the test phase, its objective is to obtain correct labels or ones as close as possible to the correct ones, not a correct relative partial order of the patterns (ranking), a total order of patterns in accordance to the order of the training set (sorting) or a total order in accordance to the training labels (multipartite ranking).

3.2.3 *Advanced related topics*

Additionally, there are other topics related to ordinal regression which have recently been studied:

- Monotonic classification [101–103] is a special class of ordinal classification task, where there are monotonicity constraints between features and decision classes, i.e. $\mathbf{x} \succeq \mathbf{x}' \rightarrow f(\mathbf{x}) \geq f(\mathbf{x}')$ [104], where $\mathbf{x} \succeq \mathbf{x}'$ means that $\mathbf{x}$ dominates $\mathbf{x}'$, i.e. $x_k \geq x'_k, 1 \leq k \leq K$. Monotonic classification tasks are very common in real-world life [102] (for example, consider the case where employers must select their employees based on their education and experience), where monotonicity may also be an important model requirement for justifying the decision made and it can be required both for binary or multiclass classifiers. This kind of problem has been approached, for example, by decision trees [102, 105] and rough set theory [103]. Feature selection has also been evaluated in monotonic classification [101], and a statistical framework for classification with monotonicity constraints has been recently proposed [104].

- A recent work is concerned with transductive ordinal regression [88], where a SVM model is derived to learn from a set of labelled and unlabelled patterns.

- Uncertainty has been included in ordinal regression models in two different ways. Nondeterministic ordinal classifiers (defined as those allowed to predict more than one label for some entries of an input space) are considered in [106]. In [35] a kernel model is proposed for those ordinal problems where partial class memberships probabilities are available instead of crisp labels.

- One step forward [107] considers those problems where the prediction labels follow a circular order (e.g. directional predictions).

All these works are outside the scope of this thesis.

3.2.4 *Ordinal regression performance metrics*

In this thesis, we mainly utilize four evaluation metrics quantifying the accuracy of N predicted ordinal labels for a given dataset $\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N\}$, with respect to the true targets $\{y_1, y_2, \ldots, y_N\}$:

- *Acc*: the accuracy (*Acc*), also known as Correct Classification Rate, is the rate of correctly classified patterns:

$$Acc = \frac{1}{N}\sum_{i=1}^{N}[\![\hat{y}_i = y_i]\!],$$

  where y_i is the true label, $\hat{y}_i$ is the predicted label and $[\![c]\!]$ is the indicator function, being equal to 1 if c is true, and to 0 otherwise. *Acc* values range from 0 to 1 and they represent a global performance on the classification task. Although *Acc* is widely used in classification tasks, is it not suitable for some type of problems, such as imbalanced datasets [108] (very different number of patterns for each class) or ordinal datasets [23]. The *Acc* can be expressed as an error, the Mean Zero-one Error (*MZE*), which is the fraction of incorrect predictions on individual samples:

$$MZE = \frac{1}{N}\sum_{i=1}^{N} I\left(\hat{y}_i \neq y_i\right).$$

On the other hand, there are other product-moment ordinal metrics specifically used in ordinal classification:

- *MAE*: The Mean Absolute Error (*MAE*) is the average absolute deviation of the predicted labels from the true labels [23]:

$$MAE = \frac{1}{N}\sum_{i=1}^{N} e(\mathbf{x}_i),$$

  where $e(\mathbf{x}_i) = |\mathcal{O}(y_i) - \mathcal{O}(\hat{y}_i)|$. The *MAE* values range from 0 to $Q-1$. Since *Acc* does not reflect the category order, *MAE* is typically used in the ordinal classification literature together with *Acc* [18, 30, 33, 41, 109, 110]. However, neither *Acc* nor *MAE* are suitable for problems with imbalanced classes. This is rectified e.g. in the *average MAE* (*AMAE*) [23] measuring the mean performance of the classifier across all classes.

- *AMAE*: This measure evaluates the mean of the *MAE*s across classes [23]. It has been proposed as a more robust alternative to *MAE* for imbalanced datasets – a very common situation in ordinal classification, where extreme classes (associated with rare situations) tend to be less populated. AMAE is defined as:

$$AMAE = \frac{1}{Q}\sum_{j=1}^{Q} MAE_j = \frac{1}{Q}\sum_{j=1}^{Q}\frac{1}{n_j}\sum_{i=1}^{n_j} e(\mathbf{x}_i),$$

  where *AMAE* values range from 0 to $Q-1$ and n_j is the number of patterns in class j.

Finally association metrics are presented, which are also used in ordinal classification:

- r_S: The Spearman's rank correlation coefficient is a non-parametric measure of statistical dependence between two variables [111]:

$$r_S = \frac{\sum_{i=1}^{N}(\mathcal{O}(y_i) - \overline{\mathcal{O}(y)})(\mathcal{O}(\hat{y}_i) - \overline{\mathcal{O}(\hat{y})})}{\sqrt{\sum_{i=1}^{N}(\mathcal{O}(y_i) - \overline{\mathcal{O}(y)})^2}\sqrt{\sum_{i=1}^{N}(\mathcal{O}(\hat{y}_i) - \overline{\mathcal{O}(\hat{y})})^2}},$$

  where $\overline{\mathcal{O}(y)}$ and $\overline{\mathcal{O}(\hat{y})}$ are the average of $\mathcal{O}(y_i)$ and $\mathcal{O}(\hat{y}_i)$, $i = 1, ..., N$, respectively. Recall that $\mathcal{O}(C_q) = q$. r_S values range from -1 to 1.

- *WKappa*: The Weighted Kappa is a modified version of the Kappa statistic calculated to allow assigning different weights to different levels of aggregation between two variables [112]:

$$WKappa = \frac{p_{o(w)} - p_{e(w)}}{1 - p_{e(w)}},$$

  where

$$p_{o(w)} = \frac{1}{N}\sum_{q=1}^{Q}\sum_{j=1}^{Q} w_{qj} n_{qj},$$

  and

$$p_{e(w)} = \frac{1}{N^2}\sum_{q=1}^{Q}\sum_{j=1}^{Q} w_{qj} n_{q\bullet} n_{\bullet j},$$

  where the weight $w_{qj} = |q - j|$ quantifies the degree of discrepancy between the true (q) and the predicted (j) categories, and *WKappa* values range from -1 to 1.

- τ_b: The Kendall's τ_b is a statistic used to measure the association between two measured quantities. Specifically, it is a measure of the rank correlation [113]:

$$\tau_b = \frac{\sum_{i,j=1}^{N} \hat{c}_{ij} c_{ij}}{\sqrt{\sum_{i,j=1}^{N} \hat{c}_{ij}^2 \sum_{i,j=1}^{N} c_{ij}^2}},$$

  where $\hat{c}_{ij}$ is $+1$ if $\hat{y}_i$ is greater than (in the ordinal scale) $\hat{y}_j$, 0 if $\hat{y}_i$ and $\hat{y}_j$ are the same, and -1 if $\hat{y}_i$ is lower than $\hat{y}_j$, and the same for c_{ij} (using y_i and y_j). τ_b values range from -1 (maximum disagreement between the prediction and the true label), to 0 (no correlation between them) and to 1 (maximum agreement). τ_b has been advocated as a better measure for ordinal variables because it is independent of the values used to represent classes [114] since it works directly on the set of pairs corresponding to different observations.

While the τ_b metric is independent on the values chosen for the ranks that represent the classes, *MAE*, *AMAE*, r_S and *WKappa* depend on the distance between ranking of two consecutive classes. However, for the association metrics (r_S, *WKappa* and τ_b), one may argue that shifting the predictions one class would will keep the same measure value whereas the quality of the ordinal classification is lower. However, note

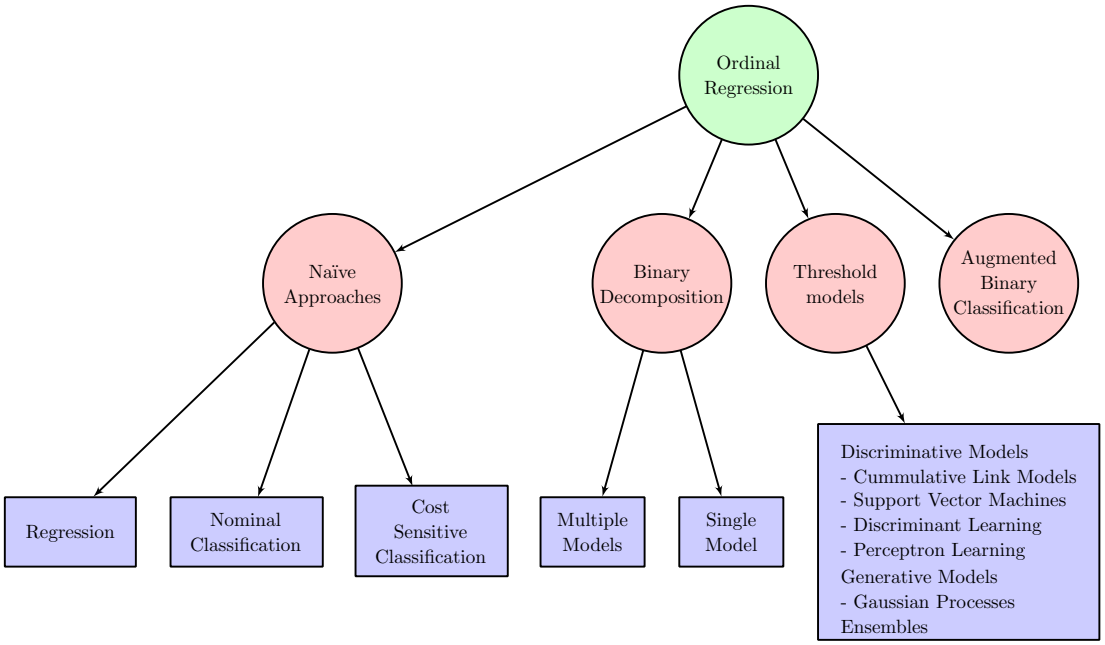


Figure 3.1: Proposed taxonomy for ordinal regression methods

that since there is a finite number of classes, shifting all predictions by one class will have detrimental effect in the boundary classes and so would substantially decrease the performance, even as measured by the association metrics. As a consequence, the association metrics are still an interesting measure for ordinal classification but should be used in conjunction with other ones.

3.3 AN ORDINAL REGRESSION TAXONOMY

In this section, a taxonomy of ordinal regression methods is proposed. With this purpose, we firstly review what have been referred to as naïve approaches, in the sense that the model is obtained by using other standard machine learning prediction algorithms (e.g. nominal classification or standard regression). Then, binary decomposition approaches are reviewed, the main idea being to decompose the ordinal problem into several binary ones, which are separately solved by multiple or a single model. The next group will include the set of methods known as threshold models, which are based on the general idea of approximating a real value predictor and then dividing the real line into different intervals. Finally, augmented binary classification techniques decompose the ordinal regression problem into a single binary classification one, but extending the dataset with new input variables and weights. The taxonomy proposed is given in Figure 3.1.

3.3.1 *Naïve approaches*

Ordinal regression problems can be easily simplified into other standard problems, which generally involves making some assumptions. As will be later discussed, these methods can be very competitive given that, even though these assumptions may not

hold, the methods inherit the performance of very well-tuned models. The following methods can be found in the literature:

3.3.1.1 *Regression*

One very simple idea to tackle ordinal regression is to cast all the different labels $\{C_1, C_2, \ldots, C_Q\}$ into real values $\{r_1, r_2, \ldots, r_Q\}$ [115], where $r_i \in \mathbb{R}$, and then to apply standard regression techniques [9, 116, 117] (least square regression, neural networks, support vector regression...). Typically, the value of each label is related to its position in the ordinal scale, i.e. $r_i = i$. For example, Kramer et al. [25, 118] map the ordinal scale by assigning numerical values, applying a regression tree model and rounding the results for assigning the class when predicting new values. They also evaluate the possibility of using the median, the mode, or the rounded mean of all the patterns in the leaves of the tree. The main problem with these approaches is that real values used for the labels may hinder the performance of the regression algorithms, and there is no principled way of deciding the value a label should have without prior information about the problem, since the distance between classes is unknown. Moreover, regression learners will be more sensitive to the representation of the label rather than its ordering [119].

3.3.1.2 *Nominal classification*

Ordinal classification problems are usually considered from a standard nominal perspective (e.g. by using standard support vector classifiers [120]), and the order between classes is simply ignored. Some researchers routinely apply nominal response data analysis methods (yielding results invariant to the permutation of the categories) to both nominal and ordinal target variables alike because they are both categorical [121]. Nominal classification algorithms completely ignore the ordering of the labels by treating them as independent classes, thus requiring more training data in general [119]. The Support Vector Machines paradigm is perhaps the most common kernel learning method for statistical pattern recognition. Beyond the application of the kernel trick to allow nonlinear decision discriminants, and the slack-variables to avoid inseparability, relax the constraints and handle noisy data, the original binary SVM had to be reformulated to deal with multiclass problems [122]. The study in this chapter compares two of the most commonly used approaches for solving this problem: the *OneVsAll* and the pairwise (or *OneVsOne*) formulations.

3.3.1.3 *Cost sensitive classification*

A more advanced method that can be considered in this group is cost-sensitive learning, which is indeed closely related to ordinal regression. Many real-world applications of machine learning and data mining require the evaluation of the learned system with different costs for different types of misclassification errors [123, 124] (in fact, cost-sensitive classification can be used to express any finite-choice and bounded-loss machine learning problems [125]). This is the case with ordinal regression, although the exact costs for misclassification are generally unknown. The cost of misclassifications can be forced to be different depending on the distance between real and predicted classes, in the ordinal scale. The work of Kotsiantis and

Table 3.1: Example of different cost matrices.

Zero-one	Absolute cost	Quadratic cost
$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 4 & 9 & 16 \\ 1 & 0 & 1 & 4 & 9 \\ 4 & 1 & 0 & 1 & 4 \\ 9 & 4 & 1 & 0 & 1 \\ 16 & 9 & 4 & 1 & 0 \end{pmatrix}$

Pintelas [26] considers cost-sensitive classification to solve ordinal regression, by using absolute costs (i.e. the element c_{ij} of the cost matrix $\mathbf{C}$ is equal to the difference in the number of categories, $c_{ij} = |i - j|$). C4.5, PART and 3-NN algorithms are shown to obtain better MAE values when cost matrices are used, without harming (in fact even improving) accuracy [26]. Two cost matrices for a five class problem are given in Table 3.1, including the absolute cost matrix and that of quadratic cost ($c_{ij} = |i - j|^2$), together with a zero-one cost matrix, which is the one assumed in nominal classification. Other possibilities are to choose asymmetric costs or non-convex two-Gaussian cost [31]. Again, the main problem is that, without a priori knowledge of the problem, it is not clear which cost matrix is more suitable for the problem.

3.3.2 *Binary decompositions*

This group includes all those methods which are based on decomposing the ordinal target variable into several binary ones, which are then estimated by a single or multiple models. A summary of the decompositions is given in Table 3.2, where five classes are considered, each method generating a different decomposition matrix. Columns of the matrix correspond to the binary subproblems and rows to the role of each class for each subproblem. The symbol $+$ is associated to the positive class and the symbol $-$ to the negative one. If the class is not used in the specific binary subproblem, no symbol is included in the corresponding position. *OneVsAll* and *OveVsOne* formulations are not related to ordinal regression, but they have been included for comparison purposes. Note the high number of binary decompositions needed by *OneVsOne* (all the 2-combinations taken from Q, in this case, 10 combinations).

Two main issues have to be taken into account when analysing the methods herein presented:

- Some of them are based on the idea of training a different model for each subproblem (*multiple model* approaches), while others learn one *single model* for all the subproblems.
- Apart from defining how to decompose the problem, it is important to define a rule for predicting new patterns, once the decision values for each subproblem are obtained. For the prediction phase, the corresponding binary codes of Table 3.2 (rows representing each class) can be considered as part of the error-

Table 3.2: Binary decompositions for a 5-class ordinal problem.

Nominal decompositions	
OneVsAll	*OneVsOne*
$\begin{pmatrix} +,-,-,-,- \\ -,+,-,-,- \\ -,-,+,-,- \\ -,-,-,+,- \\ -,-,-,-,+ \end{pmatrix}$	$\begin{pmatrix} -,-,-,-, & , & , & , & , & , & , \\ +, & , & , & ,-,-,-, & , & , & , \\ & ,+, & , & ,+, & , & ,-,-, & , \\ & , & ,+, & , & ,+, & ,+, & ,-, \\ & , & , & ,+, & , & ,+, & ,+,+, \end{pmatrix}$

Ordinal decompositions			
OrderedPartitions	*OneVsNext*	*OneVsFollowers*	*OneVsPrevious*
$\begin{pmatrix} -,-,-,- \\ +,-,-,- \\ +,+,-,- \\ +,+,+,- \\ +,+,+,+ \end{pmatrix}$	$\begin{pmatrix} -, & , & , \\ +,-, & , \\ & ,+,-, \\ & , & ,+,- \\ & , & , & ,+ \end{pmatrix}$	$\begin{pmatrix} -, & , & , \\ +,-, & , \\ +,+,-, \\ +,+,+,- \\ +,+,+,+ \end{pmatrix}$	$\begin{pmatrix} +,+,+,+ \\ +,+,+,- \\ +,+,-, \\ +,-, & , \\ -, & , & , \end{pmatrix}$

correcting output codes framework [126], considering that the final predicted class is the one closest to the predicted code form by all binary responses.

Taking the first criterion into account, we have divided binary decomposition algorithms into *multiple model* and *single model* approaches.

3.3.2.1 *Multiple model approaches*

Ordinal information gives us the possibility of comparing the different labels. For a given rank k, a direct question can be the following, "is the label of pattern $\mathbf{x}$ greater than k?" [31]. This question is clearly a binary classification problem, so ordinal classification can be solved by considering each binary classification problem independently and combining the binary outputs into a label, which is the approach followed by Frank and Hall in [27] (this decomposition is called *OrderedPartitions* in Table 3.2). In their work, Frank and Hall considered C4.5 as the binary classifier and the decision of the different binary classifiers were combined by using associated probabilities $p_q = P(y \succ C_q|\mathbf{x}), 1 \leq q \leq Q-1$:

$$\begin{aligned} P(y = C_1|\mathbf{x}) &\approx 1 - p_1, \\ P(y = C_q|\mathbf{x}) &\approx p_{q-1} - p_q, \quad 2 \leq q \leq Q-1, \\ P(y = C_Q|\mathbf{x}) &\approx p_{Q-1}. \end{aligned}$$

Note that this approach may lead to negative probability estimates [29], given that binary classifiers are independently learned and nothing assures that $p_{q-1} < p_q$. When there is no need for proper probability estimations, prediction can be done by selecting the maximum.

In the work of Waegeman et al. [28], this framework is used but explicit weights over the patterns of each binary system are imposed, in such a way that errors on training objects are penalized proportionally to the absolute difference between their rank and k. Additionally, labels for the test set are obtained by combining the estimated outcomes y_q of all the $Q-1$ binary classifiers. The interpretation of these binary outcomes $y_{qi} \in \{+1,-1\}, 1 \leq q \leq (Q-1), 1 \leq i \leq N$, intuitively leads to $y_i \succ C_q$ if $y_{qi} = 1$. In this way, the rank k is assigned to pattern $\mathbf{x}_i$ so that $y_{qi} = -1, \forall q < k$, and $y_{qi} = 1, \forall q \geq k$. As stated by the authors, this strategy can result in ambiguities for some test patterns, and they should be solved by using techniques similar to those considered for multiclass classification. A very similar scheme to [28] is proposed in [77], where the weights are obtained slightly differently, and different kernels are used for the different binary classification sub-problems. The promising accuracy obtained can be related to the selection of different kernels for each subproblem. Finally, the *OrderedPartitions* decomposition is also followed by the Bayesian hierarchical experts approach in [127].

Other binary decompositions can be found in the literature. The cascade linear utility model is used in [128], which considers $Q-1$ projections, in such a way that projection q separates classes $C_1 \cup \ldots \cup C_{Q-q-1}$ from class C_{Q-q}, i.e. one class is eliminated for each projection (this is the *OneVsPrevious* decomposition in Table 3.2). The predictions are then combined by using a union utility function. Finally, binary support vector machines were also applied to ordinal regression [6] by making use of the ordinal pairwise partitioning approach [79]. This approach is composed of four different reformulations of the classical *OneVsOne* and *OneVsAll* paradigms. *OneVsNext* approach considers that each binary classifier q separates class C_q from class C_{q+1}, and *OneVsFollowers* (which is similar to the *OneVsPrevious* approach in [128] but in the opposite direction) constructs each binary classifier q for the task of separating class C_q from classes $C_{q+1} \cup \ldots \cup C_Q$. The prediction phase is then approached by examining each binary classifier in order, so that, if a model predicts that the pattern is in the class which is isolated (not grouped with other classes), then this is the class predicted by the group of classifiers. If not, the pattern is evaluated by the next classifier, until one class is predicted. This can be done in a forward manner (starting with the first model and going forwards) or in a backward manner (starting with the last one and going backwards), both possibilities being evaluated in [6]. Without prior knowledge of the problem, it is not possible to know which one of these strategies (backwards or forwards) performs better.

Finally, another possibility [76] is to derive a classifier for each class but separating the labels into groups of three classes (instead of only two) for intermediate subtasks (labels lower than C_q, label C_q, and labels higher than C_q), or two classes for the extreme ones. The objective is to incorporate the order information in the subclassification tasks. Although the decomposition for intermediate classes is not binary but ternary, this approach has been included in this group because its motivation is similar to all the aforementioned.

3.3.2.2 *Single model approaches*

Among non-parametric models, one appealing property of neural networks is that they can handle multiple responses in a seamless fashion [47]. Usually, as many

output neurons as the number of target variables are included in the output layer and targets are presented to the network in the form of vectors $\mathbf{t}_i, 1 \leq i \leq N$ (see Section 2.2.6 in Chapter 2). When applied to nominal classification, the most usual approach is to consider a 1-of-Q coding scheme [9], i.e. $\mathbf{t}_i = \{t_{i1}, \ldots, t_{iQ}\}$, $t_{iq} = 1$ if $\mathbf{x}_i$ corresponds to an example belonging to class C_q, and $t_{iq} = 0$ (or $t_{iq} = -1$), otherwise. In the ordinal regression framework, one can take the ordering information into account to design specific ordinal target coding schemes, which can improve the performance of the methods. Indeed, all the decompositions in Table 3.2 can be used to train neural networks, by taking each row as the code for the target class, $\mathbf{t}_i$, and a single model will be obtained for all related subproblems (considering that each output neuron is solving each subproblem). This can be done by assigning a number to the different symbols in Table 3.2. For sigmoidal output neurons, a 1 is assigned for positive symbols ($+$) and a 0 for negative ones ($-$). For hyperbolic functions, negative symbols are represented with a -1 and positive ones also with a 1. Those decompositions where a class is not involved should be treated as a "does not matter" condition where, whatever the output response, no error signal should be generated [129]. Again, how to perform the predictions after training the network is an open issue, and different proposals have been made.

A generalization of ordinal perceptron learning [38] in neural networks was proposed in [130]. The method is based on two main ideas: 1) the targets are coded using the *OrderedPartitions* approach; and 2) instead of using the softmax function [9] for the output nodes, a standard sigmoid function is imposed for each output node, and the category assigned to a pattern is equal to the index previous to that of the first output node whose value is smaller than a predefined threshold T (e.g., $T = 0.5$), or when no nodes are left. This method ignores inconsistencies in the predictions (i.e. a sigmoid with value higher than T after the index selected).

Extreme Learning Machines (ELMs) are single-layer feed-forward neural networks, where the hidden layer does not need to be tuned given that corresponding weights are randomly assigned. ELMs have demonstrated good scalability and generalization performance with a faster learning speed when compared to other models such as SVMs [131]. They have been adapted to ordinal regression [132], and one of the proposed ordinal ELMs also considers *OrderedPartitions* targets. Additionally, multiple models are also trained using the *OneVsOne* and the *OrderedPartitions* approaches. For the prediction phase, the loss-based decoding approach [126] is utilized, i.e. the chosen label is that which minimizes the exponential loss, $k = \arg\min_{1 \leq q \leq Q} d_{\mathrm{L}}\left(\mathbf{M}_q, \mathbf{y}(\mathbf{x})\right)$, where $\mathbf{M}_q$ is the code associated to class q (q-th row of the coding matrix), $\mathbf{y}(\mathbf{x})$ is the vector of predictions, and $d_{\mathrm{L}}\left(\mathbf{M}_q, \mathbf{y}(\mathbf{x})\right)$ is the exponential loss function:

$$d_{\mathrm{L}}\left(\mathbf{M}_q, \mathbf{y}(\mathbf{x})\right) = \sum_{i=1}^{Q} \exp\left(-\mathbf{M}_{qi} \cdot y_i(\mathbf{x})\right).$$

The values of the vector $\mathbf{y}(\mathbf{x})$ are assumed to be in the $[-1, +1]$ range, and those of $\mathbf{M}_q$ in the set $\{-1, 0, +1\}$. The single ELM was found to obtain slightly better generalization results and also to report the lowest computational time [132]. The ELM algorithm is covered with more details in Chapter 4, and its evolutionary version is extended in several ways for dealing with imbalance issues also in Chapter 4. Moreover the ordinal ELM is extended in Section 5.2 to consider the order and class imbalance issues in the training algorithm.

Costa [129] followed a probabilistic framework to propose another neural network architecture able to exploit the ordinal nature of the data. The proposal is based on the joint prediction of constrained concurrent events, which can be turned into a classification task defined in a suitable space through a "partitive approach". An appropriate entropic loss is derived for $\mathbf{P}(\mathcal{Y})$, i.e. the set of subsets of $\mathcal{Y}$, where $\mathcal{Y}$ is a set of Q elementary events. A probability for each possible subset should be estimated, leading to a total of 2^Q probabilities. However, depending on the classification problem, not all possibilities should be examined. For example, this is simplified for random variables taking values in finite ordered sets (i.e. ordinal regression), as well as in the case of independent boolean random variables (i.e. nominal classification). To adapt neural networks to the ordinal case structure, targets are reformulated following the *OneVsFollowers* approach and the prediction phase is now accomplished by considering that, under its constrained entropic loss formulation, the output of the q-th output neuron estimates the probability that q and $q-1$ events are both true. This methodology was further evaluated and compared in other works [18, 29, 133].

Although all these neural network approaches consist of a single model, they are trained independently in the sense that the output of one neuron does not depend on the others (only on common nonlinear transformations of the inputs). That is the reason why we have included them into the category of binary decompositions.

3.3.3 *Threshold models*

Often, in the ordinal regression paradigm, it is natural to assume that an unobserved continuous variable underlies the ordinal response variable. Such a variable is called a latent variable. Latent variable models or threshold models are probably the most important type of ordinal regression models [33, 35, 134, 135]. These models consider the ordinal scale as the result of coarse measurements of a continuous variable, called the latent variable. It is typically assumed that the latent variable is difficult to measure or cannot be observed itself [35]. The threshold model can be represented with the following general expression:

$$f(\mathbf{x}, \boldsymbol{\theta}) = \begin{cases} C_1, & \text{if } g(\mathbf{x}) \leq \theta_1, \\ C_2, & \text{if } \theta_1 < g(\mathbf{x}) \leq \theta_2, \\ \vdots & \\ C_Q, & \text{if } g(\mathbf{x}) > \theta_{Q-1}, \end{cases} \tag{3.1}$$

where $g : \mathcal{X} \to \mathbb{R}$ is the function that projects data space onto the 1-dimensional latent space $\mathcal{Z} \subseteq \mathbb{R}$ and $\theta_1 \leq\leq \theta_2 \ldots \leq \theta_{Q-1}$ are the thresholds that divide the space into ordered intervals corresponding to the classes.

Then, these methodologies estimate:

- A function $f(\mathbf{x})$ that tries to predict the nature of those underlying real-valued outcomes.
- A set of thresholds $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_{Q-1}) \in \mathbb{R}^{Q-1}$ to represent intervals in the range of $f(\mathbf{x})$, which must satisfy the constraints $\theta_1 \leq \theta_2 \leq \ldots \leq \theta_{Q-1}$.

Generally, these models simultaneously estimate a projection function (similar to the ranking function to be learned by multipartite algorithms) and the best thresholds

for the ordinal classification task considered. However some methods, such as the proposed in Chapter 5 do not include the thresholds in the optimization problem, but they are adjusted and fixed independently.

3.3.3.1 *Discriminative Models*

Discriminative models estimate directly the posterior $P(y|\mathbf{x})$, or learn a function to map the input $\mathbf{x}$ to class labels.

CUMULATIVE LINK MODELS. Arising from a statistical background, the Proportional Odds Model (POM) is one of the first models specifically designed for ordinal regression [36], dated back to 1980. It is a member of a wider family of models recognized as Cumulative Link Models (CLMs) [136]. In order to extend binary logistic regression to ordinal regression, CLMs predict probabilities of group of contiguous categories, taking the ordinal scale into account. In this way, cumulative probabilities $P(y \preceq C_j|\mathbf{x})$ are estimated, which can be directly related to standard probabilities:

$$P(y \preceq C_q|\mathbf{x}) = P(y = C_1|\mathbf{x}) + \ldots + P(y = C_q|\mathbf{x}),$$
$$P(y = C_q|\mathbf{x}) = P(y \preceq C_q|\mathbf{x}) - P(y \preceq C_{q-1}|\mathbf{x}),$$

with $1 \leq q \leq Q$ and considering by definition that $P(y \preceq C_Q|\mathbf{x}) = 1$. Stochastic ordering of space $\mathcal{X}$ is satisfied by the following general model form [37]:

$$g^{-1}\left(P(y \preceq C_q|\mathbf{x})\right) = \theta_q - \mathbf{w}^T\mathbf{x}, 1 \leq q \leq Q,$$

where $g^{-1} : [0,1] \rightarrow (-\infty, +\infty)$ is a monotonic function often referred to as the inverse link function and θ_q is the threshold defined for class C_q. This model is clearly inspired by the latent variable motivation, considering that $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ is a linear transformation. A decision rule $f : \mathcal{X} \rightarrow \mathcal{Y}$ is not fitted directly. If the ordinal response is a coarsely measured latent continuous variable $f(\mathbf{x})$, label C_q in the training set is observed if and only if $f(\mathbf{x}) \in [\theta_{q-1}, \theta_q]$, where the function f (latent utility) and $\boldsymbol{\theta} = (\theta_0, \theta_1, ..., \theta_{Q-1}, \theta_Q)$ are to be determined from the data. It is assumed that $\theta_0 = -\infty$ and $\theta_Q = +\infty$, so the real line, defined by $f(\mathbf{x}), \mathbf{x} \in \mathcal{X}$, is divided into Q consecutive intervals. Each region separated by two consecutive biases corresponds to a category C_q. The constraints $\theta_1 \leq \theta_2 \leq \ldots \leq \theta_{Q-1}$ ensure that $P(y \preceq C_q|\mathbf{x})$ increases with q [137].

Suppose a model of the latent variable, $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + \epsilon$, where ϵ is the random component with zero expectation, $\mathbf{E}[\epsilon] = 0$, distributed according to F_ϵ. If a distribution assumption F_ϵ is made for ϵ, the cumulative model is obtained by choosing the inverse distribution F_ϵ^{-1} as the inverse link function g^{-1}. The most common choice for the distribution of ϵ is the logistic function (which is indeed the one selected for the POM [138]), although probit, complementary log-log, negative log-log or cauchit functions could also be used [136]. As will be seen, all the models in this section are inspired by the POM in the strategy assumed, obtaining a projection and dividing this projection into different ordered intervals, which are associated with ordered categories. This projection can be used to obtain more information about the confidence of the predictions by relating it to its proximity to the biases. Additionally, the POM model provides us with a solid probabilistic interpretation.

Focusing on the POM model [138], the distribution of ϵ is assumed to be the standard logistic function:

$$g^{-1}\left(P(y \preceq C_q|\mathbf{x})\right) = \ln\left(\frac{P(y \preceq C_q|\mathbf{x})}{P(y \succ C_q|\mathbf{x})}\right) = \theta_q - \mathbf{w}^T\mathbf{x},$$
$$odds(y \preceq C_q|\mathbf{x}) = \exp(\theta_q - \mathbf{w}^T\mathbf{x}),$$

for $1 \leq q \leq Q-1$. Consequently, the ratio of the odds for two different pattens $\mathbf{x}_0$ and $\mathbf{x}_1$ are proportional:

$$\frac{odds(y \preceq C_q|\mathbf{x}_1)}{odds(y \preceq C_q|\mathbf{x}_0)} = \exp(-\mathbf{w}^T(\mathbf{x}_1 - \mathbf{x}_0)).$$

More flexible non-proportional alternatives have been developed, one of them simply assuming different $\mathbf{w}$ for each class (which is known as the generalized ordered logit model [139]), and another applying the proportional odds assumption only to a subset of variables (partial proportional odds [140]). This second model is extended in a recent work [86], allowing the effects of a subset of variables to vary across threshold equations by a common factor. Moreover, Tutz [141] presented a general framework for parametric models that extends generalized additive models to incorporate non-parametric parts, which are fitted by maximizing penalized log-likelihood.

Another main problem with linear CLMs is that they are rather inflexible since the decision functions are always linear hyperplanes, this generally affecting the performance of the model. A nonlinear version of the POM model was proposed in [81, 137] by simply setting the projection $f(\mathbf{x})$ to be the output of a neural network. The probabilistic interpretation of CLMs can be used to apply a maximum likelihood maximization for setting the network parameters. Gradient descent techniques with proper constraints for the biases serve this purpose. This nonlinear generalization of the POM model based on neural networks was considered in [142], where an evolutionary algorithm was applied to optimize all the parameters considered. Linear ordinal logistic regression was combined with nonlinear kernel machines using primal-dual relations from Nystrom sampling [143]. However, to make the computation of the model feasible, a sub-sample from the data had to be selected, which limits the applicability to those cases where there is a reasonable way to do this [143].

SUPPORT VECTOR MACHINES. Because of their good generalization performance, SVM models are maybe the most widely applied ones to ordinal regression, their structure being easily adapted to that of threshold models. The proposal of Herbrich et al. [8, 37] is the first SVM based algorithm, but they consider a pairwise approach by deriving a new dataset made up of all possible difference vectors $\mathbf{x}_{ij}^d = \mathbf{x}_i - \mathbf{x}_j$ and $y_{ij} = \text{sign}\left(\mathcal{O}(y_i) - \mathcal{O}(y_j)\right)$, with $y_i, y_j \in \{C_1, \ldots, C_Q\}$. In contrast, all the SVM pointwise approaches share the common objective of seeking $Q-1$ parallel discriminant hyperplanes, all of them represented by a common vector $\mathbf{w}$ and the scalars biases $\theta_1 \leq \ldots \leq \theta_{Q-1}$ to properly separate training data into ordered classes. In this sense, several methodologies for the computation of the pairs $(\mathbf{w}, \theta_1), \ldots, (\mathbf{w}, \theta_{Q-1})$ can be considered. The work of Shashua and Levin [144] introduced two first methods: the maximization of the margin between the closest neighbouring classes and the maximization of the sum of margins between classes. Both approaches present two main

problems [29]: the model is incompletely specified, because the thresholds are not uniquely defined, and they may not be properly ordered at the optimal solution, since the inequality $\theta_1 \leq \theta_2 \leq \ldots \leq \theta_{Q-1}$ is not included in the formulation.

Consequently, Chu and Keerthi [33, 39] proposed two different reformulations for the same idea, solving the problem of unordered thresholds at the solution. On the one hand, they impose explicit constraints on the optimization problem, only considering adjacent labels for threshold determination (Support Vector Ordinal Regression with explicit constraints (SVOREX)). On the other hand, patterns in all the categories are allowed to contribute errors for each hyperplane (Support Vector Ordinal Regression with implicit constraints (SVORIM)), which, as they prove [33], leads to automatically satisfied constraints in the optimal solution. They empirically found that SVOREX performed better in terms of accuracy (with a more local behaviour), and SVORIM preceded in terms of absolute deviations in number of classes or MAE (with a more global behaviour), and this is justified theoretically based on the loss minimized for each method. The framework of reduction [31] also explains this from the point of view of the cost matrices selected. Their idea was reformulated in [145], as the search of concentric spheres with minimum volume, so an instance of the q-th category lies inside the sphere of radius R_q and outside the sphere of radius R_{q-1}. Generalization properties for some ordinal regression algorithms, including these SVM approaches, were further studied in [146], this study then extended by Xu et al. focusing on the strong and weak stability of ordinal regression algorithms [147].

In [148], the errors of an ordinal SVM classifier are studied separately depending on whether they correspond to upgrading errors (predicted label higher than the actual one) or downgrading ones (the predicted label being lower than the actual one). Authors address the two-objective problem of finding a classifier maximizing simultaneously the two margins, and they show that the whole set of Pareto-optimal solutions can be obtained by solving one quadratic optimization problem and then letting the thresholds vary in an appropriate range.

Some recent works focused on solving the bottleneck of these SVM proposals, which is usually the high computational complexity to handle larger datasets. Concerning this topic, two different proposals can be distinguished: block-quantized support vector ordinal regression [149] and ordinal-class core vector machines [150]. The former is based on performing kernel k-means and applying SVOR in the cluster representatives, on the idea of approximating the kernel matrix K by $\tilde{K}$ which will be composed of k^2 constant blocks, in such a way that the problem scales with the number of clusters, instead of the dataset size. The latter is an extension of core vector machines [151] in the ordinal regression setting. In that sense, the formulation of the problem as a minimum enclosing ball problem in the feature space allows linear asymptotic time complexity with the number of training patterns, while the space complexity is independent of the number of training patterns.

DISCRIMINANT LEARNING. Discriminant learning has also been reformulated to tackle ordinal regression [89]. Discriminant analysis is usually not considered as a classification technique by itself, but rather as a supervised dimensionality reduction. Nonetheless, it is widely used for that purpose, since, as a projection method, the definition of thresholds can be used to discriminate the classes. In general, to allow the computation of the optimal projection for the data, this algorithm analyses two main

objectives: the maximization of the between-class distance, and the minimization of the within-class distance, by using variance-covariance matrices and the Rayleigh coefficient. In order to reformulate the algorithm for ordinal regression, a minimum separation constraint is imposed on the closest projections of points of contiguous pairs of classes in the ordinal scale, which leads the algorithm to order projected patterns according to their label. This will preserve the ordinal information and avoid some serious ordinal misclassification errors. The methodology is known as Kernel Discriminant Learning for Ordinal Regression (KDLOR) [75, 89]. The authors claim that, compared with the SVM based methods, the KDA approach takes advantage of the global information of the data and the distribution of the classes, and also reduces the computational complexity of the problem.

The method was extended [152, 153] based on the idea of preserving the intrinsic geometry of the data in the embedded nonlinear structure, i.e. in the induced high-dimensional feature space, via kernel mapping. This consideration is the basis of manifold learning [9], and the algorithms mentioned construct a neighbourhood graph (which takes the ordinal nature of the dataset into account) and make use of a graph search algorithm for computing the shortest path between two pairs of patterns, a distance that will subsequently be used in the generalized radial basis function, thus inducing and preserving data structure in kernel mapping. A related method is proposed in [154], where several different projections are iteratively derived.

PERCEPTRON LEARNING. PRank [32, 38] is a perceptron online learning algorithm with the structure of threshold models. It was then extended by approximating the Bayes point, what provides good performance for generalization [119]. A kernelized generalization was proposed in [155], where the key difference was that the prediction problems for different patterns were coupled through the use of joint kernel functions.

3.3.3.2 *Generative Models*

Generative models learn a model of the joint probability $P(\mathbf{x}, y)$ of input patterns $\mathbf{x}$ and label y, and make the prediction by a Bayesian framework to estimate $P(y|\mathbf{x})$.

GAUSSIAN PROCESSES. Gaussian Processes for Ordinal Regression (GPOR) [41] models the latent variable $f(\mathbf{x})$ using Gaussian Processes, to estimate then all the parameters by means of a Bayesian framework. The values of the latent function $f(\mathbf{x}_i)$ are assumed to be the given by random variables indexed by their input vectors in a zero-mean Gaussian process. Mercer kernel functions approximate the covariance between the functions of two input vectors. Given the latent function f, the joint probability of observing the ordinal variables is

$$P(\mathbf{D}|f) = \prod_{i=1}^{N} P(y_i|f(\mathbf{x}_i)),$$

and the Bayes theorem is applied to write the posterior probability

$$P(f|\mathbf{D}) = \frac{1}{P(\mathbf{D})} \prod_{i=1}^{N} P(y_i|f(\mathbf{x}_i))P(f).$$

A Gaussian noise with zero mean and unknown variance σ^2 is assumed for the latent functions. The normalization factor $P(\mathbf{D})$, more exactly $P(\mathbf{D}|\theta)$, is known as the evidence for the vector of hyperparameters θ and is estimated by two different approaches in the paper: a Maximum a Posteriori approach with Laplace approximation and an Expectation Propagation with variational methods.

A more general GPOR was then proposed to tackle multiclass classification problems but with a free structure of preferences over the labels [156]. A probabilistic sparse kernel model was proposed for ordinal regression in [157], where a Bayesian treatment was also employed to train the model. A prior over the weights governed by a set of hyperparameters was imposed, inspired by the well known relevance vector machine.

3.3.3.3 *Ensembles*

From a different perspective, the confidence of a binary classifier can be regarded as an ordering preference. RankBoost [100] is a boosting algorithm that constructs an ensemble of those confidence functions to form a better ordering preference. Some efforts were made to apply a similar idea for ordinal regression problems, deriving into OR.Boost [158]. The corresponding thresholded-ensemble models inherit the good properties of ensembles, including more stable predictions and sufficient power for approximating complicated target functions [159]. The model is composed of confidence functions, and their weighted linear combination is used as the projection $f(\mathbf{x})$. Large margin bounds of the error were also obtained [158].

The previous ensembles are based on using binary classifiers with confidence values as the base learners. With a different perspective, the well-known AdaBoost algorithm was recently extended to improve any base ordinal regression algorithm [160]. The extension, AdaBoost.OR, proved to inherit the good properties of AdaBoost, improving both the training and test performance of existing ordinal classifiers.

3.3.4 *Augmented binary classification*

Although the approaches in Subsection 3.3.2 are simple to implement, their generalization performance cannot be analysed easily. The two methods included in this subsection work in different ways.

A reduction framework (RED) can be found in the works of Li and Lin [30], Lin and Li [31], where ordinal regression is reduced to binary classification, applying three main steps:

1. Given a coding matrix $\mathbf{M}$ of $(Q-1)$ rows, input patterns $(\mathbf{x}_i, y_i)$ are transformed into extended binary patterns by replicating them, $(\mathbf{x}_i^{(q)}, y_i^{(q)})$, with:

$$\mathbf{x}_i^{(q)} = (\mathbf{x}_i, \mathbf{m}_q),\ y_i^{(q)} = 2[\![q > \mathcal{O}(y_i)]\!] - 1,$$

where $1 \leq q \leq Q-1$, $\mathbf{m}_q$ is the q-th row of $\mathbf{M}$ and $[\![\cdot]\!]$ is a Boolean test which is 1 if the inner condition is true, and 0 otherwise. $Q-1$ replicates of each pattern are generated with the following weights:

$$w_{i,q} = (Q-1) \cdot |C_{\mathcal{O}(y_i),q} - C_{\mathcal{O}(y_i),q+1}|,$$

Table 3.3: Extended binary transformation for three given patterns $(\mathbf{x}_1, y_1 = C_1), (\mathbf{x}_2, y_2 = C_2), (\mathbf{x}_3, y_3 = C_3)$, the identity coding matrix and the quadratic cost matrix.

			$\mathbf{x}_i^{(q)}$		
i	q	$w_{i,q}$	$\mathbf{x}$	$\mathbf{m}_q$	$y_i^{(q)}$
1	1	$2 \cdot \lvert 0-1 \rvert = 2$	$\mathbf{x}_1$	$\{0,1\}$	$2[\![1 < 1]\!] - 1 = -1$
1	2	$2 \cdot \lvert 1-4 \rvert = 3$	$\mathbf{x}_1$	$\{1,0\}$	$2[\![2 < 1]\!] - 1 = -1$
2	1	$2 \cdot \lvert 1-0 \rvert = 2$	$\mathbf{x}_2$	$\{0,1\}$	$2[\![1 < 2]\!] - 1 = +1$
2	2	$2 \cdot \lvert 0-1 \rvert = 2$	$\mathbf{x}_2$	$\{1,0\}$	$2[\![2 < 2]\!] - 1 = -1$
3	1	$2 \cdot \lvert 4-1 \rvert = 3$	$\mathbf{x}_3$	$\{0,1\}$	$2[\![1 < 3]\!] - 1 = +1$
3	2	$2 \cdot \lvert 1-0 \rvert = 2$	$\mathbf{x}_3$	$\{1,0\}$	$2[\![2 < 3]\!] - 1 = +1$

where $1 \leq i \leq N$, C is a V-shaped cost matrix, i.e.:

$$\begin{cases} C_{\mathcal{O}(y_i),q-1} \geq C_{\mathcal{O}(y_i),q} & \text{if } \ q \leq \mathcal{O}(y_i) \\ C_{\mathcal{O}(y_i),q} \leq C_{\mathcal{O}(y_i),q+1} & \text{if } \ q \geq \mathcal{O}(y_i) \end{cases}.$$

The cost matrix must be defined a priori. An example of this transformation is given in Table 3.3.

2. A single binary classifier with confidence outputs, $f(\mathbf{x}, \mathbf{m}_q)$, is trained for the new extended patterns, aiming at a low weighted $0/1$ loss.

3. A classification rule like the following is used to construct a final prediction for new patterns:

$$r(\mathbf{x}) = 1 + \sum_{q=1}^{Q-1} [\![f(\mathbf{x}, \mathbf{m}_q) > 0]\!]. \tag{3.2}$$

All the binary classification problems are solved jointly by computing a single binary classifier. The most striking characteristic of this algorithm is that it unifies many existing ordinal regression algorithms [31], such as the perceptron ones [32], kernel ranking [94], AdaBoost.OR [160], ORBoost-LR and ORBoost-All thresholded ensemble models [158], CLM [136] or several ordinal SVM proposals (oSVM [29], SVORIM and SVOREX [33]). Moreover, it is important to highlight the theoretical guarantees provided by the framework, including the derived cost and regret bounds and the proof of equivalence between ordinal regression and binary classification. An extension of this reduction framework was proposed in [161], where ordinal regression is proved to be equivalent to a regular multiclass classification whose distribution is changed. This extension is free of the following restrictions: target functions should be rank-monotonic; and rows of loss matrix are convex.

The data replication method of Cardoso et al. [29] (whose previous linear version appeared in [4]) is a very similar framework, except that it essentially considers the absolute cost, consequently being less flexible. However, for ordinal regression, increasing the error with the absolute difference between the predicted and estimated

labels is a natural choice in the absence of any other information [81]. Another difference is that the framework of data replication includes a parameter s which limits the number of adjacent classes considered, in such a way that the replicate q is constructed by using the $q - s$ classes to its 'left' and the $q + s$ classes to its 'right' [29]. This parameter $s \in \{1, ..., Q - 1\}$ plays the role of controlling the increase of data points.

3.3.5 *Other approaches and problem formulations*

This subsection includes some methods that are difficult to consider in the previous groups. For example, an alternative methodology is proposed by da Costa et al. [18, 133] for training ordinal regression models. The main assumption of their proposal is that the random variable class associated with a given pattern should follow a unimodal distribution. For this purpose, they provide two possible implementations: a parametric one, where a specific discrete distribution is assumed and the associated free parameters are estimated by a neural network; and a non-parametric one, where no distribution is assumed but the error function is modified to avoid errors from distant classes. The same idea was then applied to SVMs in [162] by solving an ordinal problem through a single optimisation process (the all-at-once strategy).

Ordinal decision trees have mainly been applied in the context of monotonic classification [102, 105], which imposes monotonicity constraints on the classification rule. However, in [163], both decision trees [117] and nearest neighbour (NN) classifiers [164] are applied to ordinal regression problems by introducing the notion of consistency: a small change in the input data should not lead to a 'big jump' in the output decision, i.e. adjacent decision regions should have equal or consecutive labels. This rationale was used as a post-processing mechanism of a standard decision tree and as a pre- or post- processing step for the NN method. An improvement was presented in [165] to reduce the over-regularised decision region artifact by using ensemble learning techniques.

Two ordinal learning vector quantization schemes, with metric learning, specifically designed for classifying data items into ordered classes, are introduced in [34, 166]. The methods use the order information during training, both in the selection of the prototypes and for determining the way they are updated.

3.4 ORDINAL CLASSIFICATION METHODS USED FOR THE EXPERIMENTS

In this thesis we use a set of ordinal classification methods for comparison purposes. We have selected these methods because they are the most extended ones, and they have competitive classification performance. Also, there are public available implementations of the methods provided by the authors, with the exception of KDLOR, which implementation is the one provided by [75]. In the experiments in this thesis, all the kernel methods are set up with the Gaussian kernel. Here there is a summary of the ordinal classification methods, as well as their basic configuration, that are used for the experiments:

- *Gaussian Processes for Ordinal Regression* (GPOR)[1] by Chu and Ghahramani [41], presents a probabilistic kernel approach to ordinal regression based on Gaussian processes where a threshold model that generalizes the *probit* function is used as the likelihood function for ordinal variables. In addition, Chu applies the automatic relevance determination (ARD) method proposed by [167] and [168] to the GPOR model. When using GPOR with ARD feature selection, we will refer the algorithm to as GPOR-ARD.

- *Support Vector Ordinal Regression* (SVOR)[2] by Chu and Keerthi [33, 39], proposes two new support vector approaches for ordinal regression: SVOREX and SVORIM.

- RED-SVM[3], by Li and Lin [30], Lin and Li [31], applies the reduction from cost-sensitive ordinal ranking to weighted binary classification (RED) framework to SVM. In this thesis experiments, the coding matrix considered is the identity and the cost matrix is the absolute value matrix, applied to the standard binary soft-margin SVM.

- *A Simple Approach to Ordinal Regression* (ASAOR)[4] by Frank and Hall [27] is a general method that enables standard classification algorithms to make the use of order information in attributes. Here, the C4.5 method available in Weka [66] is used as the underlying classification algorithm since this is the one initially employed by the authors of ASAOR. In this way, the algorithm is identified as ASAOR(C4.5).

- The *Proportional Odds Model* (POM) is one of the first models specifically designed for ordinal regression [36]. For the POM model, the `mnrfit` function of Matlab software has been used.

- *Kernel Discriminant Learning for Ordinal Regression* (KDLOR) [89] extends the Kernel Discriminant Analysis (KDA) using a rank constraint.

- *Support Vector Machines* (SVM)[5] [44, 58] nominal classifier is included in the experiments in order to establish a baseline nominal performance. *C*-Support Vector Classification (SVC) available in libSVM 3.0 [169] is used as the SVM classifier implementation. In order to deal with the multiclass case, a "one-versus-one" approach has been considered, following the recommendations of Hsu and Lin [122].

1 GPOR, `http://www.gatsby.ucl.ac.uk/chuwei/ordinalregression.html`
2 SVOREX and SVORIM, `http://www.gatsby.ucl.ac.uk/chuwei/svor.htm`
3 RED-SVM, `http://home.caltech.edu/htlin/program/libsvm/`
4 ASAOR is available in Weka `http://www.cs.waikato.ac.nz/ml/weka/`
5 `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Part II

PROPOSALS FOR ORDINAL REGRESSION AND IMBALANCED DATA

This part of the thesis presents the main contributions to the state of the art. Chapter 4 deals with improvements regarding class imbalance, while Chapter 5 presents the major contributions to ordinal regression problem.

4

NEW PROPOSALS FOR CLASS IMBALANCE PROBLEM

Summary. This chapter deals with data imbalance issue, a very common problem in ordinal regression datasets. Data imbalance refers to datasets where the number of patterns belonging to each class varies noticeably. For classes with small number of patterns, general purpose classifiers tend to ignore minority classes and consequently misclassifying patterns as neighbour (majority) classes. In the recent years, and specially in the nominal multi-class field, this has motivated research related to produce classifiers sensitive to all the classes, as well as to develop performance evaluation metrics that consider per-class classification throughput.

Recently, a multi-objective Minimum Sensitivity-Accuracy based methodology has been proposed for building classifiers for imbalanced multi-class problems [21]. With this goal, the proposal extends a Pareto based evolutionary algorithm. However, the well known high computational cost of multi-objective approaches motivates the design of more efficient alternatives. This chapter presents an effective and efficient alternative to the Pareto based solution when considering both Accuracy and Minimum Sensitivity in multi-class classifiers. Several alternatives are presented and evaluated in the context of imbalanced datasets.

Associated publications. Some portions of this chapter appeared in [108, 170, 171]:

- J. Sánchez-Monedero, P. A. Gutiérrez, F. Fernández-Navarro, and C. Hervás-Martínez. Weighting efficient accuracy and minimum sensitivity for evolving multi-class classifiers. *Neural Processing Letters*, 34 (2): 101–116, 2011.
  JCR: 0.750
  http://dx.doi.org/10.1007/s11063-011-9186-9.
- J. Sánchez-Monedero, C. Hervás-Martínez, P.A. Gutiérrez, M. Carbonero-Ruz, M. C. Ramírez-Moreno and M. Cruz-Ramírez. Evaluating the Performance of Evolutionary Extreme Learning Machines by a Combination of Sensitivity and Accuracy Measures. *Neural Network World*, 20: 899–912, 2010.
  JCR: 0.511
  http://www.nnw.cz/obsahy10.html.
- J. Sánchez-Monedero, C. Hervás-Martínez, F. Martínez-Estudillo, Carbonero-Ruz, M. C. Ramírez-Moreno and M. Cruz-Ramírez. Evolutionary learning using a sensitivity-accuracy approach for classifica-

tion. In *Hybrid Artificial Intelligence Systems*, volume 6077 of *Lecture Notes in Computer Science*, pages 288–295. Springer Berlin / Heidelberg, 2010.
http://dx.doi.org/10.1007/978-3-642-13803-4_36.

4.1 MOTIVATION AND OBJECTIVES

Recently, a multi-objective Minimum Sensitivity-Accuracy based methodology has been proposed for building class imbalance aware classifiers [21]. The problem has been formulated as a Pareto front evolutionary strategy for evolving artificial neural networks [172]. However, the problem of such Pareto based solutions is that those algorithms are computationally expensive (see Section 4.5). With this motivation, we propose an algorithm level solution to the class imbalance problem based on a dynamic fitness function which is optimized in the learning process to balance global and per-class performance. This fitness function reformulates the multi-objective Pareto approach as a weighted linear combination of the objectives, which is a more efficient approach.

On the other hand, we also care of the learning algorithm optimization. Huang et al. have proposed an algorithm called Extreme Learning Machine (ELM) [173] which randomly chooses weights connecting input layer nodes and hidden layer nodes and analytically determines (by using Moore-Penrose generalized inverse [174, 175]) the weights connecting the hidden layer to the output layer of the network. The algorithm tends to provide good testing performance at an extremely fast learning speed. However, ELM may need a higher number of hidden nodes due to the random determination of the weights and biases between input and hidden layers. In [176], a hybrid algorithm called Evolutionary Extreme Learning Machine (E-ELM) was proposed by using the differential evolution (DE) algorithm [177]. The experimental results obtained show that this approach reduces the number of hidden nodes and obtains more compact networks.

In this chapter, the simultaneous optimization of Accuracy (or C) and Minimum Sensitivity (MS) is carried out by means of a modification of the E-ELM algorithm. The key point of this modification is the fitness function considered, which tries to take into account both C and MS objectives. A convex linear combination of both tries to achieve a good balance between the classification rate level in the global dataset and an acceptable level for each class.

The chapter follows with an introduction to the class imbalance problem (Section 4.2), the associated specific performance metrics (Section 4.3) and a taxonomy of solutions to the problem (Section 4.4). Then, a brief exposition of related multi-objective research, ELM and E-ELM works is done at Section 4.5. Section 4.6 proposes different fitness functions and the Evolutionary ELM considering C and MS (E-ELM-CS) algorithm, as well as an ELM model extension to obtain probabilities associated to the model predictions. Section 4.7 shows experiments evaluating the incidence of the λ hyper-parameter to E-ELM-CS, and compares several proposed fitness functions and different related state of the art methods in order to choose the best options considering classification performance and computational time. Finally, some conclusions are drawn.

4.2 INTRODUCTION TO CLASS IMBALANCE

A dataset is considered to be imbalanced or to present class imbalance if at least one of the classes (often called *minority class*) contains a much smaller number of patterns than the other classes (usually referred to as *majority classes*). This can be also expressed as a dataset where there are differences in class prior probabilities. In recent years, the imbalanced learning problem has drawn a significant amount of interest in machine learning [178–181]. The fundamental issue with the imbalanced learning problem is the ability of imbalanced data to significantly compromise the performance of most standard learning algorithms [181], whose performance tend to be biased to the majority classes. This problem is even more severe if we consider that typically, in real world problems, the minority class is of primary interest in a given application [182, 183]. An example of this are the majority of medical diagnosis problems, such as donor-recipient matching in liver transplantation [184]. Other real problems examples presenting class imbalance are detection of oil spills in satellite radar images [185], detection of fraudulent calls [186], risk management [187], predictive microbiology [188] and text classification [189].

However, it should be noticed than class imbalance alone is not the only dataset feature that hinders the performance of some standard classifiers. Prati et al. [190] highlight that often this performance decay has been exclusively attributed to imbalance, conversely, class imbalance does not hinder performance by itself. This deduction is straightforward since minority well separated classes are easy to classify independent of the class prior. Prati et al. [190] conclude that class overlapping have a role even stronger than class imbalance. This claim is related to the study of Murphey et al. [191], that assert that the imbalanced problem is more serious when the dataset has a high level of noise, which is highly related to class overlapping. This issue has been pointed out also by other authors [192], as well as the problem of the effect of small sample size, that is worsen by the increment of data dimensionality.

4.3 EVALUATION OF IMBALANCED PROBLEMS

Evaluation metrics are essential in machine learning. They are used to evaluate and guide the learning algorithms. In the case of imbalanced data sets, if the metric does not value the minority class, the learning algorithm will not properly be able to handle the imbalance problem [193]. This section is devoted to revise the most commonly used performance metrics for the imbalance case, along with the proposal used for the contributions of the rest of the chapter.

4.3.1 *Performance evaluation in binary imbalanced problems*

As previously mentioned, classification metrics such as Accuracy (Acc) may lead to erroneous conclusions since the majority class (or classes) have more impact on the performance metrics and the learning algorithm than the minority class (or classes), then alternative metrics are needed [194]. In binary classification, there are three alternatives which are the most commonly used [193]. Table 4.1 presents a binary classification confusion matrix, which is the base of many metrics.

Table 4.1: Contingency or confusion matrix presenting the relation between true positives, true negatives, false positives, and false negatives

	actual class (observation)	
predicted class (expectation)	TP (true positive) Correct result	FP (false positive) Unexpected result
	FN (false negative) Missing result	TN (true negative) Correct absence of result

4.3.1.1 *Metrics when the minority class is more relevant*

The first metric set is the *Precision*, *Recall* and *F-measure*, which is used when the performance of the positive class (the minority class) is considered, i.e. it is more important for the problem, in the learning task. This typically the case of medical diagnosis applications.

The *Precision* of a classifier is the percentage of positive predictions that are correct:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{4.1}$$

Recall (also *Sensitivity (S)* or true positive rate) is the rate of true positive patterns rightly classified:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{4.2}$$

The *F-measure* is related to a high value of both Precision and Recall, and is defined as the harmonic mean of both [195]:

$$\text{F-measure} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}. \tag{4.3}$$

4.3.1.2 *Metrics when both classes are relevant*

When the performance of both classes is important and is expected to be high simultaneously, the *Sensitivity*, *Specificity* and the geometric mean of both are used.

The *S* (also positive rate, or the recall rate) measures the rate of actual positives which are correctly identified as such (e.g. the percentage of people having an illness who are correctly identified as having the illness):

$$\text{Sensitivity} = \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{4.4}$$

The *Specificity* (also called positive predictive value, precision rate or True Negative Rate (TNR)) measures the proportion of negatives which are correctly identified as such (e.g. the percentage of people that do not have the illness and are correctly identified as not having it). The Specificity is the True Negative Rate (TNR):

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}. \tag{4.5}$$

Finally, Kubat and Matwin [196] proposed the geometric mean (G-mean) of the Sensitivity (precision on the positive patterns) and Specificity (precision on the negative examples):

$$\text{G-Mean} = \sqrt{\text{Sensitivity} \times \text{Specificity}}. \tag{4.6}$$

Then, the G-Mean indicates the balance between classification performances on the majority and minority classes, and it has been used by several authors [180, 185, 197]. This metric can be extended to the multiclass case by using the performance of each individual class as terms of the product in Equation 4.6.

Finally, for ending this subsection, it is worthwhile to formulate Acc in terms of the variables of the confusion or contingency matrix in Table 4.1:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4.7}$$

4.3.1.3 *ROC and AUC*

In a Sensitivity-Specificity analysis there is usually desired a trade-off between these measures. Though the use of the geometric mean of the true rates is very extended for considering the whole performance [180, 185, 197], probably the Receiver Operating Characteristic (ROC) and the area under the ROC curve (AUC) are the two most common measures for assessing the overall classification performance compromise [194].

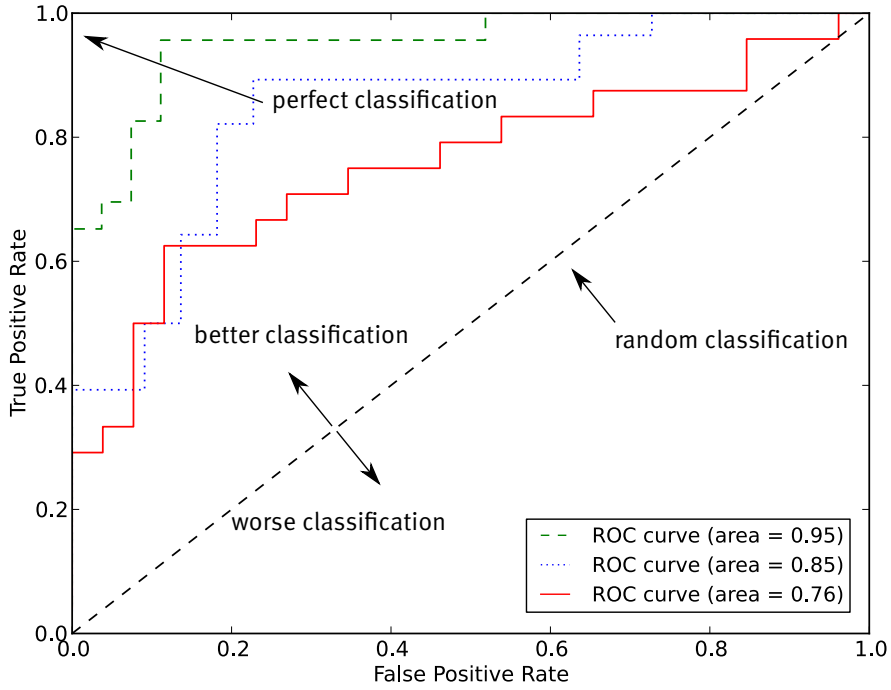


Figure 4.1: Receiver operating characteristic example comparing three classifiers. Observe the optimal classification (0.0,1.0) point, and the line indicating random classifiers.

The ROC is a two-dimensions graph of the relationship between classification achievements and classification costs in terms of the decision threshold variation. The ROC curve reveals that the true positive rate cannot increase without also increasing

the false positive rate. The aforementioned trade-off can be graphically shown as a ROC curve, allowing visual comparison of different classifiers performance.

Moreover to the visual comparison, it might be necessary to obtain a numerical representation of a classifier performance (e.g. for using the metric to guide a optimization search). The ROC performance can be reduced to a single scalar value representing expected performance, normally calculating the AUC which measures the misclassification rate of one class and the accuracy of the other. Then the AUC has the attractive property that it side-steps the need to specify the costs of the different kinds of misclassification [198].

Since the AUC is a portion of the area of the unit square, its value will always be between 0 and 1. However, because random guessing produces the diagonal line between $(0,0)$ and $(1,1)$, which has an area of 0.5, no realistic classifier should have an AUC less than 0.5 [195]. The AUC has the following statistical property: the AUC of a classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance [195]. Finally, it should be noticed than if AUC < 0.5 it can be stated than the classifier has worse performance than a *random classifier*. See Figure 4.1 for ROC curves and AUC values examples.

The AUC presents two limitations. The first one is the fastening to binary problems, though some attempts have being introduced in the late years to extend ROC analysis to multi-class problems. The second limitation of the AUC evaluation is that it depends on a classifier output providing patterns' scores to allow patterns ordering. However many classifiers do not provide this type of output[1]. If this is the case, the geometric mean approach is a reasonable alternative.

4.3.2 *Performance evaluation in multi-class imbalanced problems*

The ROC curve and the AUC have been widely used to analyze and evaluate the quality of binary classifiers [200, 201] however its application to the multi-class case is not straightforward. Srinivasan [202] has shown that, theoretically, the ROC analysis extends to more than two classes "directly" [203]. In addition, different approaches have been proposed to extend the ROC analysis to multi-class classification [198, 203, 204]. More precisely, the ROC analysis in the Q class case ($Q > 2$) implies the calculation of the Volume Under the ROC Surface (VUS). Nevertheless this calculation is not trivial and there are computational limits to its estimation. However, not only there are computational limitations but also representational ones. ROC analysis in two dimensions has a very nice and understandable representation, but it cannot be directly extended to more than two classes, because even for 3 classes we have a 6D space [203].

Other proposals, such as the one of Hand and Till [198], are valid for non-direct multi-class formulation, i.e. one-versus-one and one-versus-all strategies. However, these proposals are linked to the underlying multi-class decomposition scheme, so

1 It should be noticed that there is an active work in extending classifiers for this purpose. For instance, though neural networks models do not typically produce scores or probabilistic outputs, it is possible to obtain a probabilistic output adding a softmax [199] transformation layer to the neural network model output layer. However, other probabilistic extensions are not so direct.

that a direct multi-class evaluation can be more suitable and can be better extended to different types of classifiers.

Due to the lack of a robust and efficient alternative to AUC for multi-class, the geometric mean of the true rates is probably the most extended metric [180, 183, 185, 197] class imbalance. However, lately it has been proposed the aforementioned Minimum Sensitivity-Accuracy pair with the purpose of visualizing and evaluating multi-class classifiers. The explanation of this proposal is done in the next subsection.

4.3.3 *Accuracy and Minimum Sensitivity*

As exposed in the current chapter, in nominal classification, to evaluate a classifier, the machine learning community has traditionally used the Accuracy to measure its default performance. In the same way, Accuracy, as other global performance metrics such as the Mean Squared Error (MSE), has been frequently used as the fitness function in evolutionary algorithms when solving classification problems. However, the pitfalls of using Accuracy have been pointed out by several authors [205] since it only reflects one-dimensional – the global performance – aspect of the classification.

Martínez-Estudillo et al. [206] address the problem of the one dimensional consideration in multi-class problems. In that work, two measures are considered to evaluate a classifier: traditionally used Accuracy (*Acc*, *CCR* or *C*) and the proposed minimum sensitivities of all classes, i.e. Minimum Sensitivity; that is, the premise assumed is that a good classifier should combine a high correct classification rate level in the generalization set with an acceptable accuracy level for each class. The following exposition is a summary of the works of Martínez-Estudillo et al. [206] and Fernández-Caballero et al. [21], for further details please check these references.

A classification problem with Q classes and N training or testing patterns is considered, with g as a classifier obtaining a $Q \times Q$ contingency or confusion matrix $M(g) = \left\{ n_{ij};\ \sum_{i,j=1}^{Q} n_{ij} = N \right\}$, where n_{ij} represents the number of times the patterns are predicted by classifier g to be in class j when they really belong to class i. The main diagonal corresponds to the correctly classified patterns and the off-diagonal to the mistakes in the classification task.

The number of patterns associated with class i can be denoted by $n_{i\bullet} = \sum_{j=1}^{Q} n_{ij}$, $i = 1, \ldots, Q$. Two scalar measures are derived, which take the elements of the confusion matrix into consideration from different points of view [21, 206]. Let $S_i = n_{ii}/n_{i\bullet}$ be the number of patterns correctly predicted to be in class i with respect to the total number of patterns in i (Sensitivity for class i). Therefore, the Sensitivity for class i estimates the probability of correctly predicting a class i example. From the above quantities, the Minimum Sensitivity (MS) of the classifier is defined as the minimum value of the sensitivities for each class, $MS = \min\{S_i;\ i = 1, \ldots, Q\}$. Moreover, the Correct Classification Rate or Accuracy is the rate of all the correct predictions:

$$C = (1/N) \sum_{j=1}^{Q} n_{jj}. \tag{4.8}$$

The two-dimensional measure (MS, C) associated to a classifier g is an interesting alternative for representing its behaviour (MS on the horizontal axis and C on the vertical axis). A classifier depicted in this space is giving information about two of

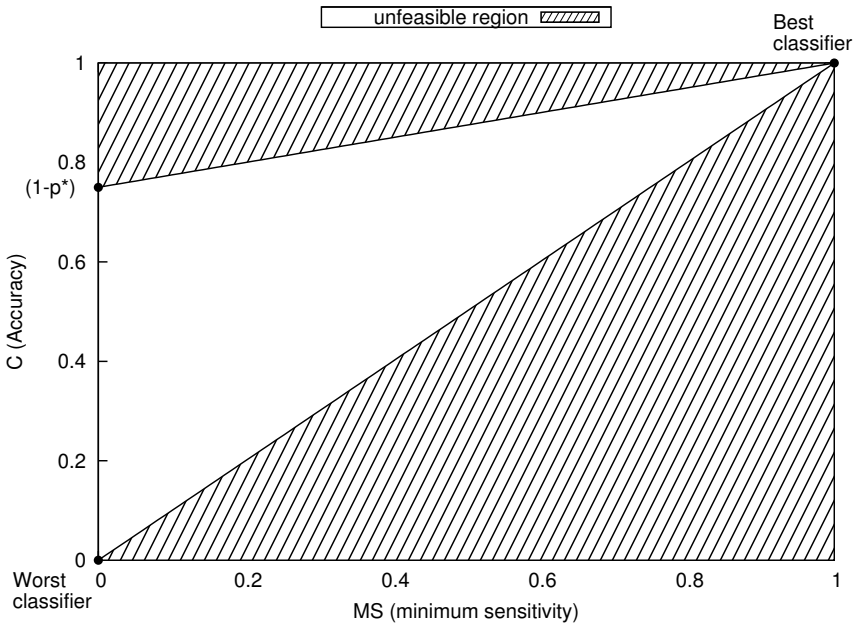


Figure 4.2: Unfeasible region in the two-dimensional (MS, C) space for a concrete classification problem.

its features: the global performance and the performance in each class. One point in (MS, C) space dominates another if it is above and to the right, i.e., it has more Accuracy and greater Sensitivity.

It is straightforward to prove the following relationship between C and MS (see [206]). Let us consider a Q-class classification problem. Let C and MS associated with a classifier g, then $MS \leq C \leq 1 - (1 - MS)\, p^*$, where $p^* = n_{q\bullet}/N$ is the minimum of the estimated prior probabilities (q is the class with less number of patterns).

Therefore, each classifier will be represented as a point outside the shaded region in Figure 4.2. Several points in (MS, C) space are important to note. The lower left point $(0, 0)$ represents the worst classifier and the optimum classifier is located at the $(1, 1)$ point. Furthermore, the points on the vertical axis correspond to classifiers that are not able to predict any point in a concrete class correctly. Note that it is possible to find among them classifiers with a high level of C, particularly in problems with small p^*.

The optimization strategies in the (MS, C) space should try to move the classifiers towards the optimum classifier located in the $(1, 1)$.

4.4 SOLUTIONS FOR THE CLASS IMBALANCED PROBLEM

The previous section has introduced and motivated the problem research, and it has preliminary depicted some associated issues. The present section introduced how the class imbalance problem is addressed, which is currently done from two points of view that are not exclusive:

DATA PREPROCESSING LEVEL. The data is preprocessed in order to lighten the class distribution imbalance. These techniques are known as *resampling techniques*, and the purpose is to change the classes distribution by suppressing or adding

patterns. Resampling techniques can be divided into three groups: *undersampling* methods, which delete instances (typically from the majority classes); *oversampling* methods, which create a superset of the original dataset by replicating some instances or creating new synthetic instances from existing ones; and finally, *hybrid* methods, that combine both sampling approaches [192]. Probably the most extended oversampling technique is the SMOTE (Synthetic minority over-sampling technique) [182].

MODEL AND ALGORITHM LEVEL. Many existing classifiers' models and/or training algorithms can be modified for dealing with data imbalance. For instance, cost-sensitive learning takes into account the variable cost of a misclassification of the different classes [207], for instance adding more penalty to false negative errors in binary classification (i.e. misclassifying the interest class). The approach of Fernández-Caballero et al. [21] can be placed in this category since the algorithm optimizes two performance metrics for considering the per-class performance. This can be viewed as a type of cost-sensitive learning in which the cost matrix does not need to be adjusted. Another example, is the threshold-moving, this is, the output decision thresholds are moved toward inexpensive classes such that examples with higher costs become harder to be misclassified [208].

Regarding the second category, since most popular classifiers are reported to be inadequate when faced with the class imbalance problem, there are several works extending those models to improve classification in this context. For instance, decision trees [178, 194, 209], Support Vector Machines [178, 210, 211], neural networks [21, 178, 183], Bayesian networks [187], nearest neighbour classifiers [187] or associative classification approaches [212], among others. Recently, Fernández-Caballero et al. [21] demonstrated that global class performance and minority class performance can be competitive objectives in classifiers training, which justified the use of a multi-objective optimization approach, and they proved it in the context of MLP training.

An example of an hybrid approach is the method of Fernández-Navarro et al. [183] in which a dynamic oversampling technique is proposed in the context of evolutionary optimization. While most resampling techniques can be considered as *static* techniques, in that dynamic approach the SMOTE algorithm is selectively applied to the worst classified classes during the evolution.

4.5 RELATED WORKS

4.5.1 *Multi-Objective Evolutionary Optimization*

As introduced in Section 4.3.3, our approach tries to build classifiers with C and MS simultaneously optimized. These objectives are not always cooperative, specially with high C and MS values [21]. Moreover, considering the multi-objective evolutionary framework, C and MS are opposite objectives at high levels. This fact justifies the use of a multi-objective framework for the learning algorithm. However, since MS is not a differentiable function, most classic optimization methods can not be applied. This second point motivates the use of evolutionary algorithms as a heuristic to op-

timize the objectives. This kind of algorithms are formally known as multi-objective evolutionary algorithms (MOEAs).

The idea of designing neural networks within a multi-objective approach was first considered by Abbass in [213, 214]. In these works, the multi-objective problem formulation essentially involved the setting up of two objectives: the complexity of the network (in terms of number of hidden nodes in the neural network) and the training error (measured with MSE). For addressing that, an algorithm called memetic Pareto artificial neural networks (MPANN) which uses Pareto differential evolution was proposed, showing improvements with respect to many other MOEAs.

Fernández et al. [21] extended the NSGA-II algorithm [215] by including C and MS as the objectives in the algorithm[2]. In addition, the NSGA-II was hybridized with iRprop$^+$ [216] as the local search procedure, but this algorithm was only applied in specific generations during the evolution, the resulting algorithm is called MPENSGA-II. This Pareto multi-objective approach has been later successfully applied for solving predictive microbiology problems [217]. These problems are often imbalanced and in general the classes with less number of patterns are the most important classes.

However, it is well known that Pareto-based approaches are expensive in terms of computational time as pointed out in Coello [218]. The main problem with Pareto ranking is that there is no efficient algorithm to check for non-dominance in a set of feasible solutions [219]. As a consequence, traditional algorithms have serious performance degradation as the size of the population, the number of generations and the number of objectives increases [218].

One alternative for addressing multi-objective problems in a more efficient strategy (in terms of computing time) is to combine objectives into a single function which is normally denominated *aggregating function* [218]. This option can be suitable when the behaviour of the objective functions is more or less well known, and it presents an unique candidate solution, that can be an advantage in some domains. The weighted sum approach is one alternative for implementing an aggregating function. This method consist on adding the different objective functions with different weights for each one of the functions, then the multi-objective problem is turned into a scalar optimization problem formulated as:

$$\min \sum_{i=1}^{k} w_i f_i(\mathbf{x}),$$

where f_i is an objective function, w_i is the weight coefficient representing the importance of f_i and $\sum_{i=1}^{k} w_i = 1$.

Weighted linear combination proves to be very efficient in practice for certain types of problems, for example in combinatorial multi-objective optimization, and its computational cost is noticeably lower than other multi-objective approaches [218]. Some of the applications of this technique are schedule evaluation of a resource scheduler or design multiplierless IIR filters [220]. The main disadvantage is that it may be difficult to determine the proper weights. However, this option is a good method when there are two objectives, as it is our case, and when the first Pareto front has a very small number of models, in some cases only one (an example of this last case can be

2 Actually in that work the cross-entropy error [16] was used as an equivalent function to C.

seen at experimental results from MPANN methodology in Balance dataset in Figure 4.6 at Subsection 4.7.2).

4.5.2 *Extreme Learning Machine and Differential Evolution*

The Extreme Learning Machine (ELM) algorithm has been proposed by Huang et al. [42, 173]. ELM and its extensions have been applied to microarray gene expression cancer diagnosis [221, 222], sales forecasting [223], real-time watermarking [224], Quality-of-Service (QoS) violation detection in multimedia transmission [225] and other problems. This section briefly presents ELM algorithm and the Evolutionary ELM.

Let us consider the training set $\mathbf{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathcal{X},\ y_i \in \mathcal{Y}, i = 1, \ldots, N\}$, where N is the number of samples, Q is the number of classes, and K the number of dimensions. Then, each pattern is represented by a K-dimensional feature vector $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^K$, i.e. $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iK})$, and a class label $y \in \mathcal{Y} = \{C_1, C_2, \ldots, C_Q\}$. Since the single model neural networks have one output neuron per class, in this case, the label is represented as a $Q \times 1$ target vector $\mathbf{t}_i \in \mathbb{R}^Q$, i.e. $\mathbf{t}_i = (t_{i1}, t_{i2}, \ldots, t_{iQ})$, where $t_{ij} = 1$ means that pattern $\mathbf{x}_i$ belong to class j and $t_{ij} = 0$ means the pattern does not belong to class j. This target coding is known as a 1-of-Q coding scheme [9].

Let us consider a MLP with M nodes in the hidden layer and Q nodes in the output layer given by:

$$f(\mathbf{x}, \theta) = (f_1(\mathbf{x}, \boldsymbol{\theta}_1), f_2(\mathbf{x}, \boldsymbol{\theta}_2), \ldots, f_Q(\mathbf{x}, \boldsymbol{\theta}_Q)), \tag{4.9}$$

where:

$$f_q(\mathbf{x}, \boldsymbol{\theta}_q) = \beta_0^q + \sum_{j=1}^{M} \beta_j^q \sigma_j(\mathbf{x}, \mathbf{w}_j), q = 1, 2, \ldots, Q, \tag{4.10}$$

where $\theta = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_Q)^T$ is the transpose matrix containing all the neural net weights, $\boldsymbol{\theta}_q = (\boldsymbol{\beta}^q, \mathbf{w}_1, \ldots, \mathbf{w}_M)$ is the vector of weights of the q output node, $\boldsymbol{\beta}^q = \beta_0^q, \beta_1^q, \ldots, \beta_M^q$ is the vector of weights of the connections between the hidden layer and the qth output node, $\mathbf{w}_j = (w_{1j}, \ldots, w_{Kj})$ is the vector of weights of the connections between the input layer and the jth hidden node, Q is the number of classes in the problem, M is the number of sigmoidal units in the hidden layer and $\sigma_j(\mathbf{x}, \mathbf{w}_j)$ the sigmoidal function:

$$\sigma_j(\mathbf{x}, \mathbf{w}_j) = \frac{1}{1 + \exp\left(-\left(w_{0j} + \sum_{i=1}^{K} w_{ij} x_i\right)\right)},$$

where w_{0j} is the bias of the jth hidden node.

The linear system $f(\mathbf{x}_j) = \mathbf{t}_j, j = 1, 2, \ldots, N$, can be written as the following matrix system $\mathbf{H}\boldsymbol{\beta} = \mathbf{T}$, where $\mathbf{H}$ is the hidden layer output matrix of the network:

$$H(\mathbf{x}_1, \ldots, \mathbf{x}_N, \mathbf{w}_1, \ldots, \mathbf{w}_M) = \begin{bmatrix} \sigma(\mathbf{x}_1, \mathbf{w}_1) & \cdots & \sigma(\mathbf{x}_1, \mathbf{w}_M) \\ \vdots & \ddots & \vdots \\ \sigma(\mathbf{x}_N, \mathbf{w}_1) & \cdots & \sigma(\mathbf{x}_N, \mathbf{w}_M) \end{bmatrix}_{N \times M},$$

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1 \\ \vdots \\ \boldsymbol{\beta}_M \end{bmatrix}_{M \times Q} \quad \text{and} \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_N \end{bmatrix}_{N \times Q}.$$

The ELM algorithm randomly selects the $\mathbf{w}_j = (w_{1j}, \ldots, w_{Kj}), j = 1, \ldots, M$, weights and biases for hidden nodes, and analytically determines the output weights $\beta_0^q, \beta_1^q, \ldots, \beta_M^q$ for $q = 1 \ldots Q$ by finding the least square solution to the given linear system. The minimum norm least-square solution (LS) to the linear system is $\hat{\boldsymbol{\beta}} = \mathbf{H}^\dagger \mathbf{T}$, where $\mathbf{H}^\dagger$ is the Moore-Penrose generalized inverse [174, 175] of matrix $\mathbf{H}$. The minimum norm LS solution is unique and has the smallest norm among all the LS solutions, which guarantees better generalization performance.

However, ELM may need a higher number of hidden nodes due to the random determination of the input weights and hidden biases. The Evolutionary Extreme Learning Machine (E-ELM) [176] improves the original ELM by using a differential evolution algorithm so that more compact networks can be obtained. DE was proposed by Storn and Price [177] and it is known as one of the most efficient evolutionary algorithms which many applications such as artificial neural networks training [226]. The E-ELM uses DE to select the input weights between input and hidden layers and the Moore-Penrose generalized inverse to analytically determine the output weights between hidden and output layers. A non-evolutionary alternative to obtain simpler ELM models is the Optimally Pruned Extreme Learning Machine (OP-ELM) by Miche et al. [227]. The OP-ELM performs network pruning by applying the Multiresponse Sparse Regression (MRSR) to rank hidden layer neurons together with a leave-one-out (LOO) validation strategy.

For ending this section we want to summarize the controversy generated by the ELM term. In 2008, Wang and Wan [228] presented an article entitled *Comments on "The extreme learning machine"* to the IEEE Transactions on Neural Networks journal, the abstract is the following:

> This comment letter points out that the essence of the "extreme learning machine (ELM)" recently appeared has been proposed earlier by Broomhead and Lowe and Pao et al., and discussed by other authors. Hence, it is not necessary to introduce a new name "ELM".

The author or ELM responded with a letter explaining the differences between previous works and "ELM" in a complementary article (see Reply to "Comments on "The extreme learning machine"" [229]). Nevertheless the controversy, the motivation to research in efficient non-iterative learning methods have attacked the community and the number of works related to ELM has been growing in the last years [230].

4.6 PROPOSAL 1: EVOLUTIONARY ELM CONSIDERING ACCURACY AND MINIMUM SENSITIVITY

4.6.1 *Fitness function design*

As mentioned at the motivation section, our approach tries to build classifiers with simultaneously optimized C and MS. Since these objectives are not always cooperative [21, 206], neither MS is differentiable, an evolutionary multi-objective approach could be used.

In this work, a linear combination of these objectives is used to obtain the maximization of objectives C and MS.

We assume we do not have a priori information about the proper weighting of C and MS for each dataset. Thus, both measures are considered equally important. Thereby, we deal with this problem by adapting the algorithm to each dataset through a nested cross-validation procedure. Our purpose is to design a fitness function able to weight up both C and MS objectives in the algorithm.

There is no rule for establishing priorities between C and MS. Thus, we include a parameter for weighing the two objectives. The underlying idea is to have an automatically adjustable fitness function which could be optimized for each dataset via this parameter, called λ, ranging between $[0, 1]$. In this work, three fitness functions are proposed to try to balance the two objectives.

The first fitness function is based on C and MS. This function evaluates the performance of a classifier depending on a weighted Accuracy level and a weighted Minimum Sensitivity. It is defined by:

$$F_{\lambda CS} = (1 - \lambda)C + \lambda MS. \tag{4.11}$$

The below function is a direct expression of the objectives that can be useful for studying the algorithm behaviour in the (MS, C) space, hereafter the use of this function for fitness evaluation in an algorithm can not be optimal. According to [9, 16], in general terms, the use of continuous function for training neural networks for classification problems makes the convergence of the algorithm more robust. Then, by using, for instance, the root mean square error (RMSE) or the cross-entropy error [16] the fitness function is turned into a continuous function.

In order to properly calculate the RMSE and the cross-entropy error, the neural network outputs need to be interpreted as probabilities p_q, thereby, they must satisfy the following constraints [16]:

$$\sum_{q=1}^{Q} p_q(\mathbf{x}, \boldsymbol{\theta}_q) = 1, \tag{4.12}$$

$$0 \leq p_q(\mathbf{x}, \boldsymbol{\theta}_q) \leq 1. \tag{4.13}$$

The first constraint also ensures that the distribution is correctly normalized, so that $\int p(\mathbf{t}|\mathbf{x})d\mathbf{t} = 1$. These constraints can be satisfied by choosing a p_q output to be related to corresponding network outputs f_q by a softmax function [199]. Then, the softmax activation function is added to standard ELM model outputs:

$$p_q = p_q(\mathbf{x}, \boldsymbol{\theta}_q) = \frac{\exp(f_q(\mathbf{x}, \boldsymbol{\theta}_q))}{\sum_{i=1}^{Q} \exp(f_i(\mathbf{x}, \boldsymbol{\theta}_i))}, 1 \leq q \leq Q, \tag{4.14}$$

where f_q are the ELM outputs defined in Equation 4.10 and p_q is the posterior probability that a pattern $\mathbf{x}$ has of belonging to class q. A pattern will belong to the class with the greatest membership probability, this is:

$$C(\boldsymbol{\theta}_q, \mathbf{x}) = \arg\max_q p_q(\mathbf{x}, \boldsymbol{\theta}_q), 1 \leq q \leq Q. \tag{4.15}$$

Then, once a probabilistic function from the ELM output is determined, a fitness function based on RMSE is proposed:

$$F_{\lambda R}(\theta) = (1-\lambda)\frac{1}{1+\frac{1}{N}\sum_{q=1}^{Q}(n_q R_q(\theta))} + \lambda\frac{1}{1+\max\{R_q(\theta), q=1,\ldots,Q\}}, \quad (4.16)$$

where $n_q = \sum_{q=1}^{Q} n_{iq}, i = 1,\ldots,Q$ is the number of patterns associated with class q. $R_q(\theta)$ is the RMSE per class in the problem, defined by:

$$R_q(\theta) = \sqrt{\frac{\sum_{i=1}^{n_q}\sum_{q=1}^{Q}(t_i^q - p_i^q(\mathbf{x}_i,\theta)))^2}{n_q Q}}, \quad (4.17)$$

where N is the number of patterns, t_i^q is the target value for class q of pattern $\mathbf{x}_i$ (t_i^q will be equal to 1, in terms of probability, if the pattern $\mathbf{x}_i$ belongs to class q and 0 otherwise), p_i^q is the probability pattern $\mathbf{x}_i$ has of belonging to class q, and n_q is the number of patterns associated with class q.

The fitness function defined in Eq. 4.16 introduces an alternative for considering C and MS. The first term represents the global accuracy error for all the classes while the second term represents the isolated error of the worst classified class as defined in Eq. 4.17. The maximum error is selected because it is the equivalent to consider Minimum Sensitivity, which is the minimum Accuracy for each class.

The third fitness function proposed is based on cross-entropy error in a similar way to $F_{\lambda R}$ defined in Eq. 4.16:

$$F_{\lambda E} = (1-\lambda)\frac{1}{1+\frac{1}{N}\sum_{q=1}^{Q}(n_q E_q)} + \lambda\frac{1}{1+\max\{E_q, q=1,\ldots,Q\}}, \quad (4.18)$$

where E_q is the cross-entropy error per class in the problem, defined by:

$$E_q(\theta) = \frac{-\sum_{i=1}^{n_q}\sum_{q=1}^{Q}(t_i^q \ln(p_q(x_i,\theta_q))}{n_q Q}, \quad (4.19)$$

This error function is also known as the negative log likelihood and, when it is minimized, maximum likelihood estimates ($p_q(\mathbf{x}_i,\theta_q)$) are obtained for the event observed.

4.6.2 *The E-ELM-CS Algorithm*

Our proposed method is implemented by using the E-ELM [176]. E-ELM for classification problems only considers the misclassification rate of the classifier. Further details about the algorithm can be consulted in [176]. The E-ELM has been extended in two ways. First, the three fitness functions designed in Section 4.6.1 are added, including the addition of the softmax layer to the model. Secondly, a 10-fold cross-validation is applied, using exclusively the training data, which aims to optimally configure the λ parameter of the fitness function. Note that after several experiments with different λ values no generic optimal value for λ was found to maximize both C and MS, that is, λ depends on the data set (see experiments at Section 4.7.1). Therefore, a cross-validation process is mandatory for each different dataset. Then, the algorithm extension is called E-ELM-CS . The E-ELM-CS algorithm pseudocode is

Require: E-ELM-CS (P (Training Patterns), T (Training Tags), F (Fitness Function), λ)

$\hat{\lambda} \leftarrow$ Calculate optimal λ for F and (P,T)
Create a random initial population $\boldsymbol{\theta} = [\mathbf{w}_1, \ldots, \mathbf{w}_k, b_1, \ldots, b_k]$ of size N
for each individual **do**
  $\hat{\boldsymbol{\beta}} \leftarrow$ ELM_output($\mathbf{w}, P, T$) {Calculate output weights}
  fitness $\leftarrow$ get_fitness($\mathbf{w}, \hat{\boldsymbol{\beta}}, F, \hat{\lambda}, P, T$) {Evaluate individual}
end for
Select best individual of initial population
while Stop condition is not met **do**
  Mutate random individuals and apply crossover as described in [176]
  for each individual in the new population **do**
    $\hat{\boldsymbol{\beta}} \leftarrow$ ELM_output($\mathbf{w}, P, T$) {Calculate output weights}
    fitness $\leftarrow$ get_fitness($\mathbf{w}, \hat{\boldsymbol{\beta}}, F, \hat{\lambda}, P, T$) {Evaluate model}
    Select new individuals for replacing individuals in old population
  end for
  Select the best model in the generation
end while
return Best ELM model

function $\hat{\boldsymbol{\beta}} = ELM\_output(\mathbf{w}, P, T)$
Calculate the hidden layer output matrix H
Calculate the output weight $\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{T}$
return $\hat{\boldsymbol{\beta}}$

function $F_\lambda = get\_fitness(\mathbf{w}, \boldsymbol{\beta}, F, \lambda, P, T)$
if $F_{\lambda CS}$ **then**
  Build training confusion matrix $\mathbf{M}$
  Calculate C and MS from $\mathbf{M}$
  Get classifier fitness with Eq. (4.11)
else if $F_{\lambda R}$ **or** $F_{\lambda E}$ **then**
  Add softmax layer to the ELM model $(\mathbf{w}, \hat{\boldsymbol{\beta}})$
  if $F_{\lambda R}$ **then**
    Get classifier fitness with Eq. (4.16)
  else
    Get classifier fitness with Eq. (4.18)
  end if
end if
return Individual fitness

Figure 4.3: E-ELM-CS algorithm pseudocode.

shown in Figure 4.3. Mutation, crossover and selection operators work as described in [176].

The cross-validation is performed by testing a range of λ values for the chosen fitness function in E-ELM-CS and a given configuration for the remaining parameters. The training set is stratified into 10 sets so 10 validation configurations can be formed.

Each one of the 10 validation tests consists of different combinations of 9 sets for training and a different one for validation. Note that generalization data is never used during this cross-validation procedure so that the final algorithm performance will be measured only with unseen data. For each λ value to validate, the E-ELM-CS algorithm is run three times with the same data partition. Therefore the total number of executions over the training data is 30. The λ considered as optimal is the one shown in the following equation:

$$\hat{\lambda} = \arg\max_{\lambda_i} \frac{\overline{C_{\lambda_i}} + \overline{MS_{\lambda_i}}}{2}, \tag{4.20}$$

where $\overline{C_{\lambda_i}}$ is the mean C and $\overline{MS_{\lambda_i}}$ is the mean MS obtained by the algorithm in the different validation folds using λ_i for the fitness function.

The parameters related to the evolutionary algorithm are the same for the whole cross-validation process with the exception of the parameter related to the number of generations, which is reduced to 1/5 of the final number of generations because, experimentally, it is not necessary to go further in the number of generations in order to find the best λ. The λ values to be tested are selected from the range $[0, 1]$ in intervals of 0.25. Previous experiments confirmed that there were no significant differences if the cross-validation was performed with more values. So, considering a few λ values and reducing the number of generations in the algorithm, the cross-validation process time is drastically reduced.

4.7 EXPERIMENTS

In this section we perform some experiments with two purposes. First, to analyse the behaviour of the E-ELM-CS algorithm regarding the weight of the two objectives (Section 4.7.1), and secondly, we analyse the performance of the three fitness functions of E-ELM-CS compared to several related methods and some baseline methods (Section 4.7.3).

The E-ELM-CS is implemented as an extension of E-ELM source code available at the authors' public website[3]. For all the experiments, the crossover and mutator parameters were set up as described in [176] (the crossover constant CR was 0.8, the constant factor F, which controls the amplification of the differential variation, was 1 and the tolerance rate was 0.02).

4.7.1 *Analysis of the effect of λ values*

In this subsection, we consider the effect of the λ values. The objective of this study is to evaluate how the E-ELM-CS can achieve very different results depending on the λ value selected, and how different datasets can demand different λ values.

Firstly to evidence that C and MS can be competitive objectives we present the evolution of E-ELM-CS across several iterations[4] for BreastC database (see Table 4.2) in Figure 4.4.

3 http://www3.ntu.edu.sg/home/egbhuang/

4 Note we present more iterations than usual here in order to better observe the behaviour of the algorithm.

Table 4.2: Datasets used for the experiments observing the influence of λ parameter.

Dataset	Size	#Input	#Classes	Distribution	p^*
BreastC	286	15	2	(201,85)	0.2972
Balance	625	4	3	(288,49,288)	0.0784
BTX	63	3	7	(9,9,9,9,9,9,9)	0.1429

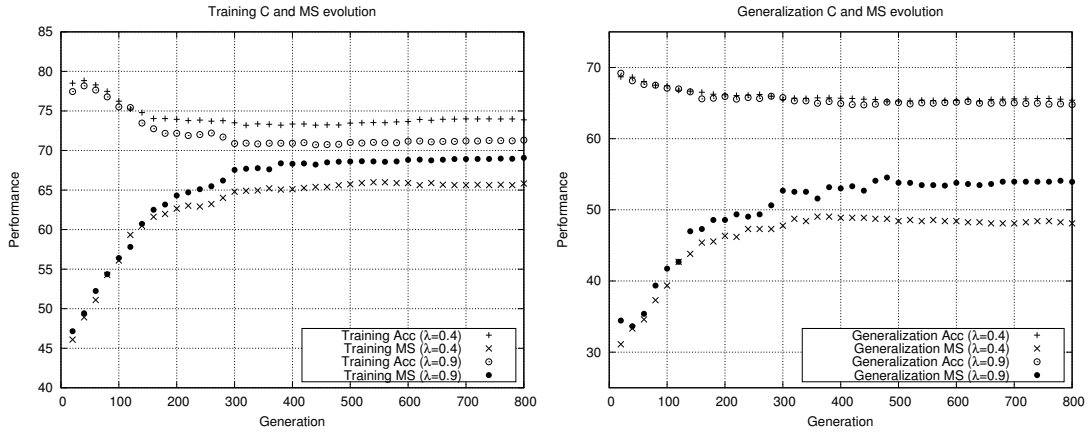
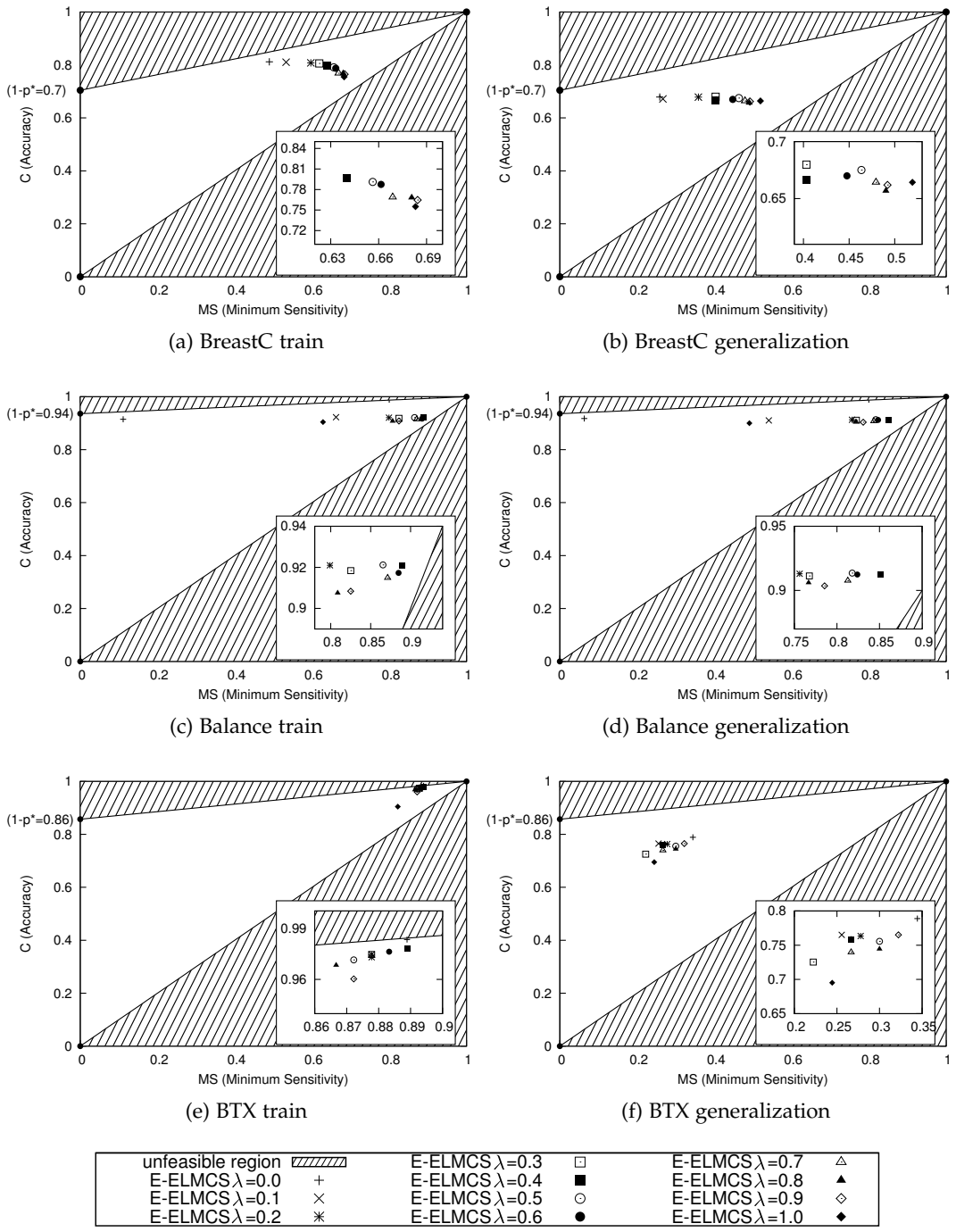


Figure 4.4: C and MS evolution for BreastC database using E-ELM-CS. The training and generalization performance of the best individual for each generation is showed. Observe than imposing more weight to the MS ($\lambda = 0.9$) move the best individual of the population though better performance in MS, but this is done at the cost of C. Even in the case of $\lambda = 0.4$, the best individual performance in C is decreased in order to increase the MS performance. However, the generalization performance of C is the same for the two λ options. This gives us a hint about the necessity of optimizing the λ value for each dataset with a validation set.

Second, we analyse the generalization performance of different λ values. Here we consider two datasets with different features taken from the UCI repository [231] and one real-world problem of analytical chemistry (benzene-toluene-xylene (BTX) and their mixtures discrimination, see Hervás-Martínez et al. [232]). Table 4.2 shows the features for each dataset. The experimental design was conducted using a stratified holdout procedure (see Prechelt [233]) with 30 runs, where approximately 75% of the patterns were randomly selected for the training set and the remaining 25% for the generalization set.

In this second series of experiments, the number of individuals in the population were 100 and the number of generations were set up to 50. The number of hidden nodes of the neural network was obtained by a cross-validation procedure varying the number of hidden nodes between 5 and 20.

In this section, we briefly observe the effect of the λ value of the fitness function $F_{\lambda CS}$ described in Eq. 4.11 on the classifier performance in terms of C and MS. For this experiments, we have chosen this function and discarted the other proposals because we can represent the training and generalization performance in the (MS, C) space described in Section 4.3.3. For BreastC, Balance and BTX datasets described in Table 4.2, both training and generalization performance results are presented. Figure



(a) BreastC train (b) BreastC generalization

(c) Balance train (d) Balance generalization

(e) BTX train (f) BTX generalization

Figure 4.5: Different λ results for BreastC, Balance and BTX databases. For each dataset the training results are placed on the left column and the generalization performance is shown in the right column. For each figure, a box containing a higher scale representation of the most interesting zone is included.

4.5 presents results of E-ELM-CS for all the λ values in the set $[0.0, 0.1, \ldots, 1.0]$. The results are the mean of the best models of 30 runs for each configuration. Each subfigure in Figure 4.5 shows a box containing a higher scale representation of the most interesting zone. Note that changing the values of λ gives us different points which are similar to the Pareto front points in multi-objective problems [218].

Subfigure 4.5a, showing BreastC performance results, clearly proves that C and MS can be competitive objectives. Observe that classifiers which are moved through higher Sensitivity values lost performance in the global classification accuracy. A trade-off point between increasing Minimum Sensitivity without loosing lot of global accuracy could be classifiers trained with $\lambda = 0.4$ or $\lambda = 0.5$. Looking at the generalization results in Subfigure 4.5b, it can be checked that the degree of over-fitting is not excessively high, and the behaviour of the fitness function for different λ values is quite similar to that in the training set.

Subfigures 4.5c and 4.5d show a very clear example of how a balance between the two objectives is necessary. The results shows that when only considering C ($\lambda = 0.0$) the method cannot improve results for all the classes. Furthermore, Subfigure 4.5d shows that using only MS ($\lambda = 1.0$) as the unique classification performance measurement is also not suitable. Then, we can consider that $\lambda = 0.4$, $\lambda = 0.5$ and $\lambda = 0.6$ have the best results for improving the two measures.

Finally, we comment the BTX performance results. In Subfigure 4.5e it can be seen that λ value has not a very significant influence on the results achieved by E-ELM-CS. However, it should be noticed that using only MS ($\lambda = 1.0$) is not suitable, and the best results are obtained by using only C ($\lambda = 0.0$). This makes sense since BTX is a perfectly balanced dataset (see number of patterns per class distribution for the BTX dataset in Table 4.2) and with a not very high noise level [232], so the behaviour of the classifiers is usually very similar for all classes.

In this preliminary analysis we can conclude that there is no rule for determining the best λ value. Therefore, we propose to optimize this parameter by the cross validation procedure described in the following section.

4.7.2 *Comparison with other evolutionary approaches*

For motivational purposes, in this section we compare the performance of E-ELM-CS algorithm with the original E-ELM algorithm (equivalent to E-ELM-CS with $\lambda = 0.0$), and two evolutionary training algorithms for MLP which are not specifically designed for dealing with class imbalance:

- MPANN [214]. MPANN is a MOEA based on differential evolution with two objectives; one is to minimize the MSE and the other is to minimize ANN complexity (in terms of the number of hidden units). The Back Propagation Algorithm [234, p. 578] is used in MPANN as local search. We have implemented a Java version using the pseudocode shown in [214] and the framework for evolutionary computation JCLEC [235]. We select both extremes of the Pareto front to compare the results with those from E-ELM-CS: the methodology is named MPANN-MSE when the extreme selected is that providing the best MSE; or it is called MPANN-HN if the extreme that is chosen has the best complexity value (number of hidden units).

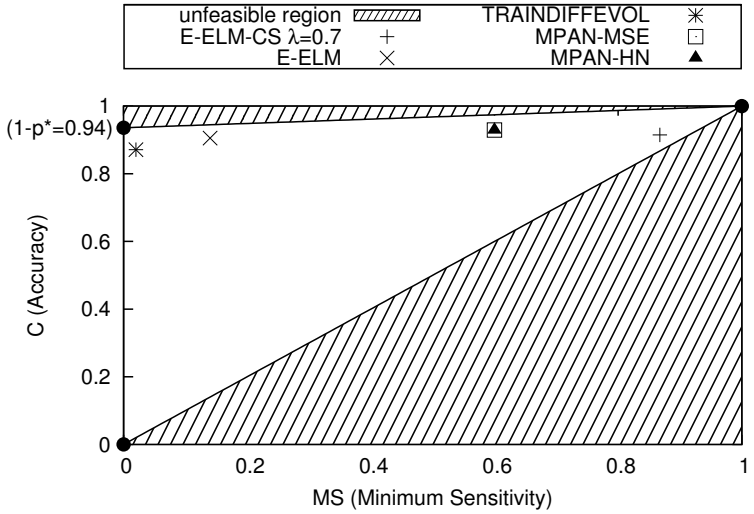


Figure 4.6: Comparison of E-ELM-CS, E-ELM, TRAINDIFFEVOL, MPANN-MSE and MPANN-HN methods for Balance database in the generalization set. Observe the performance differences between E-ELM and E-ELM-CS.

- TRAINDIFFEVOL (differential evolution training algorithm for neural networks) [226]. TRAINDIFFEVOL is an algorithm to train feed forward MLP neural networks based on differential evolution. This algorithm uses the MSE regularized by the mean squared weights and biases for guiding the networks training. To obtain the Sensitivity for each class, a modification of the source code provided by the author[5] has been implemented.

As an example of the usefulness of the (MS, C) representation, as well as the influence of the fitness function, Figure 4.6 depicts the Minimum Sensitivity-Accuracy generalization results (in mean of the results of best individuals of 30 runs) of the four evolutionary methodologies for the Balance dataset in the (MS, C) space. A visual inspection of the figure allows us to easily observe the difference in the performance of E-ELM-CS with respect to E-ELM , TRAINDIFFEVOL and MPANN.

4.7.3 *Comparison with related methods and reference classifiers*

The purpose of the experiments is to evaluate which fitness function is more suitable for E-ELM-CS with the purpose of simultaneous optimization of C and MS. Computational cost, in terms of training time T, is also considered. Results are compared with related state of the art methods.

There were ten UCI repository datasets with different features under study [231] (see Table 4.3). The experimental design was conducted using a stratified holdout procedure with 30 runs of each algorithm, where approximately 75% of the patterns were randomly selected for the training set and the remaining 25% for the generalization set. All the data have been standardized and the experiments have been conducted using Matlab R2009a running on a Ubuntu Server (x86_64 architecture) on a Intel Xeon at 2.00GHz with 8 Gb RAM.

5 http://www.it.lut.fi/project/nngenetic/

Table 4.3: Datasets used for the experiments comparing the three proposals and related methods.

Dataset	Size	#Input	#Classes	Distribution	p^*
Two classes					
BreastC-W	699	9	2	(458,241)	0.3448
Card	690	51	2	(307,383)	0.4449
Hepatitis	155	19	2	(32,123)	0.2069
Multi-class					
Balance	625	4	3	(288,49,288)	0.0784
Gene	3175	120	3	(762,765,1648)	0.2400
Iris	150	4	3	(50,50,50)	0.3333
Lymph	148	38	4	(2,81,61,4)	0.0135
Anneal	898	59	5	(8,99,684,67,40)	0.0089
Glass	214	9	6	(70,76,17,13,9,29)	0.0421
Zoo	101	16	7	(41,20,5,13,4,8,10)	0.0396

4.7.3.1 *Machine learning methods used for comparison purposes*

The experimental section compares two basically different methodologies with different extensions for training MLP neural networks. The first group of classifiers are variations of the Evolutionary ELM (results are also compared to the original ELM and OPELM [227]):

- EELM. This method is set up with two fitness functions: CCR (EELM(C)) and MS (EELM(S)).
- EELMCS. This algorithm is set up with the three fitness functions proposed in Section 4.6.1: $F_{\lambda CS}$ (EELMCS(CS)), $F_{\lambda R}$ (EELMCS(R)) and $F_{\lambda E}$ (EELMCS(E)).

The second type of neural network training algorithm is the memetic Pareto differential evolution neural network (MPDENN) presented in [236]. MPDENN is a MOEA based on the Pareto differential evolution algorithm (PDE) presented by Abbass et al. [213], Abbass [237]. The MPDENN trains ANNs considering C and MS as conflicting objectives which should be simultaneously optimized. In addition, the MPDENN applies a local search procedure to some individuals in the population. The local search algorithm used is the improved Resilient Backpropagation (iRprop$^+$) algorithm [216]. However, whereas local search can improve classification performance it is computationally costly. For this reason, MPDENN can be used without local search; when doing so, we will refer to it to as PDENN.

In addition, the experiments include two additional popular learning algorithms: a standard MLP trained with Resilient backpropagation (Rprop) algorithm [238] and the Support Vector Machine (SVM) [44, 58]. The Rprop Matlab's implementation

is used for the former algorithm, while Cost Support Vector Classificacion (SVC) available in libSVM 3.0 [169, 239] is used as the SVM classifier implementation.

All the ELM-based neural network methods were trained with different algorithms (ELM, OPELM, EELM and EELMCS) using the Sigmoid function as the basis function. For ELM and OPELM, the number of hidden nodes gradually increases in intervals of 5 within the interval $[5, 200]$ and the nearly optimal number of nodes for ELM and OPELM are then selected based on the 10-fold cross-validation method using the training set. For the E-ELM-CS(R), the range of the interval has been reduced for cross-validation of the number of hidden nodes to $[5, 20]$. PDENN and MPDENN automatically prune nodes, so a range of maximum and minimum numbers of hidden nodes must be provided. We have used the range $[1, 6]$ according to the author's recommendations. Regarding Rprop, a nested cross-validation was also performed using a range of hidden nodes of $[1, 30]$ with an interval of one. The radial basis kernel was used with the SVC method. For the selection of SVC hyperparameters (regularization parameter, C, and width of the Gaussian functions, γ), a grid search was performed with a 10-fold cross-validation, using the following ranges: $C \in \{10^{-1}, 10^{0}, \ldots, 10^{2}\}$ and $\gamma \in \{10^{-8}, 10^{-6}, \ldots, 10^{2}\}$. For all the methods, except PDENN and MPDENN, the optimal hyperparameters $\hat{\boldsymbol{\theta}}$ cross-validation criteria was the following:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}_i} \frac{\overline{C_{\boldsymbol{\theta}_i}} + \overline{MS_{\boldsymbol{\theta}_i}}}{2}. \tag{4.21}$$

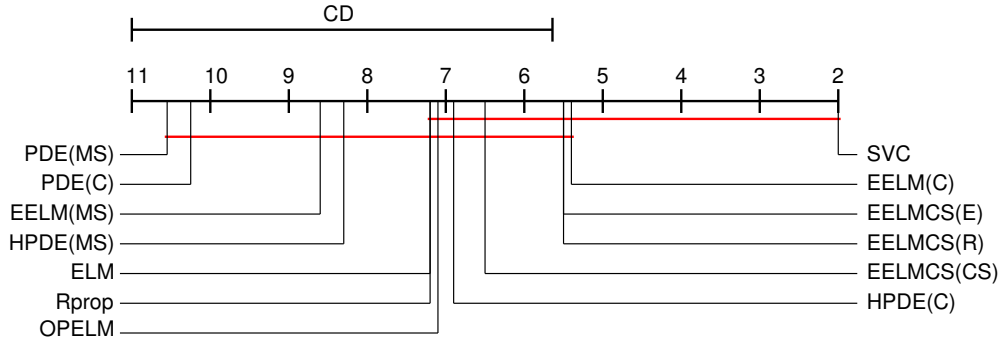
Table 4.4: Mean statistical results and average rankings

Method	$\overline{C}_{\mathrm{G}}(\%)$	$\overline{R}_{C_{\mathrm{G}}}$	$\overline{MS}_{\mathrm{G}}(\%)$	$\overline{R}_{MS_{\mathrm{G}}}$	$\overline{T}(secs.)$	$\overline{R}_{T}$
EELMCS(R)	86.19	5.50	**58.86**	**2.45**	1.42E+002	7.00
EELMCS(E)	*86.87*	5.50	56.52	4.95	1.46E+002	8.20
EELMCS(CS)	86.47	6.50	*58.50*	*4.75*	1.39E+002	7.00
EELM(C)	86.78	*5.40*	48.81	6.75	1.41E+002	7.00
EELM(MS)	84.86	8.60	57.54	5.20	1.40E+002	6.80
OPELM	85.71	7.10	42.70	10.35	3.32E+000	3.70
ELM	86.09	7.20	44.33	7.90	**9.12E–002**	*1.70*
PDE(C)	82.92	10.25	41.81	9.45	2.08E+003	10.05
PDE(MS)	82.06	10.55	51.62	7.60	2.08E+003	10.35
HPDE(C)	86.07	6.90	46.91	7.25	1.37E+006	12.25
HPDE(MS)	84.89	8.30	55.60	6.35	1.37E+006	12.35
Rprop	85.06	7.20	38.05	11.20	8.39E-001	3.30
SVC	**88.60**	**2.00**	53.97	6.80	*1.44E-001*	**1.30**

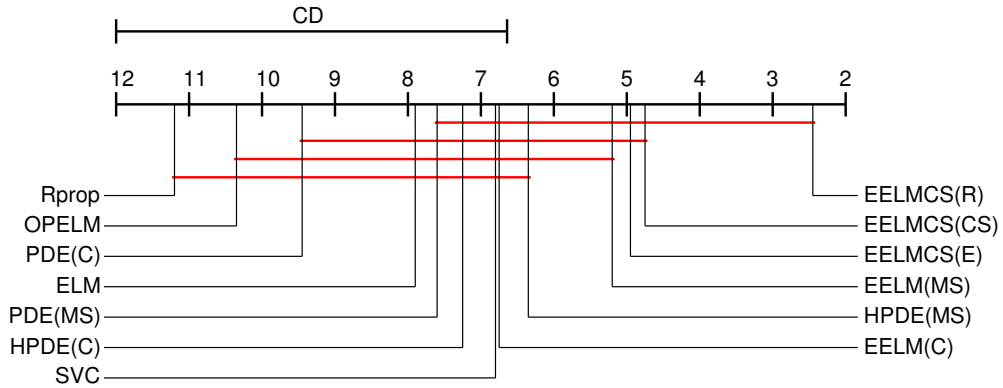
4.7.3.2 *Statistical results*

Table A.1 in Appendix A presents results in values of the mean and the standard deviation (SD) for $\%C_{\mathrm{G}}$, $\%MS_{\mathrm{G}}$ and T (training time in seconds) for 30 runs. Subindex

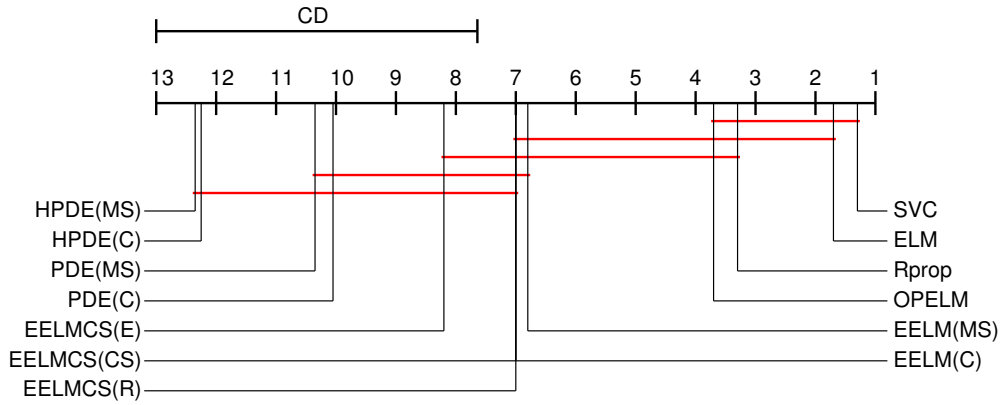
G in $\%C_G$ and $\%MS_G$ indicates that results belongs to the generalization dataset. For these tables and Table 4.4 the best result is in bold face and the second best result is in italics.



(a) Nemenyi CD diagrams comparing generalization CCR mean results of different methods.



(b) Nemenyi CD diagrams comparing the generalization MS mean results of different methods.



(c) Nemenyi CD diagrams comparing the training time mean results of different methods.

Figure 4.7: Ranking tests for CCR, MS and training time.

The mean rankings of C_G, MS_G and T are obtained to compare the different methods (see Table 4.4). A Friedman's non-parametric test for a significance level of $\alpha = 0.1$ has been carried out to determine the statistical significance of the differences in rank in each method. The test rejected the null-hypothesis stating that all algorithms performed equally in the mean ranking of C_G, MS_G and T so a Nemenyi post-hoc test [240] ($\alpha = 0.1$) was used to compare all the methods and their variations.

Table 4.5: Table with the different algorithms compared with the EELMCS(R) (i.e. the Control Algorithm) using the Holm procedure in terms of MS_G.

i	Algorithm	z	p	α'_{Holm}
1	Rprop	5.02398	0.00000	0.00833
2	OPELM	4.53594	0.00001	0.00909
3	PDE(C)	4.01918	0.00006	0.01000
4	ELM	3.12922	0.00175	0.01111
5	PDE(MS)	2.95697	0.00311	0.01250
6	HPDE(C)	2.75601	0.00585	0.01429
7	SVC	2.49764	0.01250	0.01667
8	EELM(C)	2.46893	0.01355	0.02000
9	HPDE(MS)	2.23926	0.02514	0.02500
10	EELM(MS)	1.57897	0.11434	0.03333
11	EELMCS(E)	1.43542	0.15117	0.05000
12	EELMCS(CS)	1.32059	0.18664	0.10000

Figure 4.7 shows Critical Difference (CD) diagrams proposed in [240]. Subfigure 4.7a shows that there are two groups of classifiers regarding C_G. SVC has the best C_G ranking mean but it does not have significant differences compared with EELM variants but with EELM(MS). Regarding, MS_G, all the methods but Rprop, OPELM, PDE(C) and ELM have no significant differences when they are compared to one another. Regarding T, all the non-evolutionary approaches have similar performance, in addition, EELM methods (except EELMCS(E)) show similar computational time. As expected, the Pareto based solutions have the highest computational cost (especially the hybrid approaches).

Based on the robustness regarding C_G and MS_G, and considering computational cost, EELMCS(R) promises to be the best alternative. However, when comparing them to each other for MS_G, EELMCS(R) shows no significant differences with EELM(C), so our proposal cannot be justified. From a statistical point of view, this can be explained considering the effects of the results of competitive methods, that is, the presence of EELM related methods. Therefore, the more powerful Holm post-hoc test is used to compare EELMCS(R) to all the other classifiers in order to justify our proposal. The Holm test is a multiple comparison procedure that can work with a control algorithm and compares it to the remaining methods [240]. The test statistics for comparing the i-th and the j-th method using this procedure is

$$z = \frac{R_i - R_j}{\sqrt{\frac{k(k+1)}{6N}}}, \tag{4.22}$$

where k is the number of algorithms and N the number of datasets. The z value is used to find the corresponding probability from the table of normal distribution, which is then compared to an appropriate level of confidence α. Holm's test adjusts the value for α in order to compensate for multiple comparisons.

The results of the Holm tests ($\alpha = 0.1$) for MS_G can be seen in Table 4.5, using the corresponding p and adjusted α (α'_{Holm}) values. The EELMCS(R) is used as the Control Method. The horizontal line shows the division between methods significantly different from EELMCS(R) (in terms of MS_G when $p < \alpha'_{Holm}$) and methods which are not significantly different. Considering the results of these tests, it can be concluded that the EELMCS(R) algorithm obtains a significantly higher ranking of MS_G when compared to most of the remaining methods, especially ELM, OPELM and EELM(C). Standard methods such as Rprop or SVC are not competitive when considering MS.

4.8 CONCLUSIONS

This chapter presents an efficient alternative to the Pareto based approach to train multi-class classifiers with a simultaneous improvement in C and MS.

Three different fitness functions were evaluated by extending the Evolutionary Extreme Learning Machine algorithm for training ANNs and were compared with different machine learning methodologies. Continuous fitness functions have proved to be more robust and suitable for evolutionary algorithms (see results details in Table A.1).

Statistical tests demonstrate that experimentally the E-ELM-CS(R) methodology gets similar results in CCR with respect to the other methods. However, statistical tests for MS demonstrate experimentally that it is significantly different than most algorithms. Considering the methods with the best significant results in CCR and MS, and considering the statistical test for training time T, we conclude that the weighted combination of global RMSE and the worst classified class RMSE give competitive results. Moreover of the computational time improvement, this type of alternatives present an unique candidate solution, then there is no need of further expert decisions for selecting solutions from the Pareto front.

MS results in anneal, balance, glass or hepatitis datasets (see Table Table A.1) show that our proposal is specially suitable for problems with high number of classes and/or with small size classes, i.e. imbalanced datasets. Experimental results show that some methods focus only on the bigger classes.

5

NEW PROPOSALS FOR ORDINAL REGRESSION

Summary. This chapter presents the major contributions of this thesis to the ordinal regression field. The contributions of this chapter are three.

The first contribution links the class imbalance problem and the ordinal regression problem with a proposal that considers both issues during the optimization phase.

The second contribution is the latent variable modelling based on sampling random values from several probability distributions (one for each class) in order to reconstruct the latent variable. This is the result of the initial research in modelling the ordinal latent space.

Motivated by this second contribution, an alternative approach to generate the latent variable in a more guided way is proposed, trying to exploit data ordering in the input space and mapping it to the latent variable. This is achieved by the third contribution, the proposed Pairwise Class Distance (PCD) projection which is integrated into a classification algorithm with robust results. This last contribution occupies most of the chapter.

Associated publications. Some portions of this chapter appeared in [241–243]

- J. Sánchez-Monedero, M. Carbonero-Ruz, D. Becerra-Alonso, F. Martínez-Estudillo, P.A. Gutiérrez, and C. Hervás-Martínez. Numerical variable reconstruction from ordinal categories based on probability distributions. In *Proceedings of the 11th International Conference on Intelligent Systems Design andApplications (ISDA 2011)*, pages 1182–1187, Cordoba, Spain, Spain, nov 2011.
  http://dx.doi.org/10.1109/ISDA.2011.6121819.
- J. Sánchez-Monedero, P. A. Gutiérrez, P. Tiňo, and C. Hervás-Martínez. Exploitation of pairwise class distances for ordinal classification. *Neural Computation*, Accepted: MIT Press, 2013.
  JCR: 1.884
  http://dx.doi.org/10.1162/NECO_a_00478.
- J. Sánchez-Monedero, P.A. Gutiérrez, M. Pérez-Ortiz, and C. Hervás-Martínez. An n-spheres based synthetic data generator for supervised classification. In *International Work-Conference on Artificial Neural Networks (IWANN)*, volume 7902 of *Lecture Notes in Computer Science*, pages 613–621. Springer-Verlag Berlin Heidelberg, 2013.

- J. Sánchez-Monedero, P.A. Gutiérrez and C. Hervás-Martínez. Evolutionary ordinal Extreme Learning Machine. *International Conference on Hybrid Artificial Intelligence Systems (HAIS 2013)* (Accepted).

5.1 MOTIVATION AND OBJECTIVES

In this chapter we present three new methodologies which contribute to the ordinal regression state of the art. The first proposal is the Evolutionary Extreme Learning Machine for Ordinal Regression (E-ELMOR), which adapts some of the improvements made in Chapter 4 to ordinal regression. The E-ELMOR outperforms related methods considering both magnitude of errors and class imbalance. This approach is completely covered in Section 5.2.

Apart from the first proposal, in this chapter we focus on latent variable methods, also named to as threshold methods. These ordinal classification approaches perform a projection from the input space to a one-dimensional (latent) space that is partitioned into a sequence of intervals or thresholds (one for each class). Class identity of a novel input pattern is then decided based on the interval its projection falls into. This chapter is focused on the exploration of new ways of modelling latent variable for ordinal classification. In the proposals, we reformulate the classification problem as a standard regression problem, so that class labels are turned into a continuous regression response variable. Well designed and tested regression methods can then be applied to the problem derived.

The second contribution is a latent variable modelling as a random variable that is sampled, depending on the pattern class, from a set of different probability distributions. We call this alternative Numerical Variable Reconstruction (NVR) and it is fully covered in Section 5.3. Although this proposal works well for some datasets, we observed that it was not robust enough when considering a wider range of datasets and more performance metrics.

The NVR approach does not assume any ordering in the input space, but only on the labels space, which is the strict definition of ordinal regression. In spite of this strict definition (see discussion in Section 3.2 of Chapter 3), some authors such as Hühn and Hüllermeier [17] have extended the definition of OR by suggesting that the labels ordering should be somehow present in the input space. Therefore, a motivation to develop ways of modelling the latent variable by using information about patterns distribution in the input space is found.

Most of latent variable methods train the projection as part of the overall model fitting in order to improve classification performance. However, as with any latent model fitting, direct construction hints one may have about the desired form of the latent model can prove very useful for obtaining high quality models. The key idea of the third proposal is to construct such a projection model directly, using insights about the class distribution obtained from pairwise distance calculations. This direct modelling is done trough our proposed Pairwise Class Distance (PCD) projection, and the associated classifier is called Pairwise Class Distances for Ordinal Classification (PCDOC). The approach is extensively evaluated with eight nominal and ordinal classifiers methods, ten real world ordinal classification datasets, and four different performance measures. The PCDOC methodology obtained the best results in average ranking when considering three of the four performance metrics considered,

although significant differences are found only for some of the methods. Also, after observing other methods internal behaviour in the latent space, we conclude that, for some datasets, the internal projection of those methods do not fully reflect the intra-class behaviour of the patterns. Our method is intrinsically simple, intuitive and easily understandable, yet, highly competitive with state-of-the-art approaches to ordinal classification.

Therefore, while the second proposal performs an *indirect* modelling of the latent variable, the third proposal performs a *direct* modelling of the latent variable[1].

The rest of the chapter is organized as follows. The E-ELMOR method is proposed and evaluated in Section 5.2. The NVR proposal is presented and covered with experiments in Section 5.3. Section 5.4 explains the proposed PCD data projection method and the associated classification algorithm. It also evaluates the behaviour of the projection using two synthetic datasets and the performance of the classification algorithm under situations that may hamper classification. For this purpose, an specific synthetic data generator has been developed. The section also presents experiments with real problems, and discusses the experimental results. For this chapter we will follow the mathematical notation and threshold methods definition presented in sections 3.2 and 3.3.3 of Chapter 3.

5.2 PROPOSAL 1: EVOLUTIONARY EXTREME LEARNING MACHINE FOR ORDINAL REGRESSION

In this section we propose an evolutionary Extreme Learning Machine for ordinal regression. We modify the Extreme Learning Machine for Ordinal Regression (ELMOR) model proposed by Deng et al. [132] with an extension to allow a probabilistic formulation of the neural network, for which we propose a fitness function that consider restrictions related to ordinal regression problems. We evaluate the proposal with eight datasets, five related methods and three specific performance metrics. In this section we work with the problem definition done at Section 4.5.2.

5.2.1 *ELM for Ordinal Regression*

As briefly introduced in Chapter 3, the ELM has been adapted to ordinal regression by Deng et. al [132] being the key of their approach the output coding strategies that impose the class ordering restriction. That work evaluates single multi-class and multi-model binary classifiers. The single ELM was found to obtain slightly better generalization results for benchmark datasets and also to report the lowest computational time for training. In the present work the single ELM alternative will be used. In the single ELMOR approach the output coding is the *OrderedPartitions* targets binary decomposition by Frank and Hall [27] (see Section 3.3.2.1 in Chapter 3), an example of five classes ($Q = 5$) decomposition is shown in Table 5.1.

In this way, the solutions provided by the $\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{T}$ expression (see Section 4.5.2) tend to produce order aware models. For the generalization phase, the loss-based

1 Note that if we strictly follow the statistical definition of *latent variable*, that type of variables cannot be directly modelled. However we keep the term *latent variable* because of the relation of the method to other ordinal regression methods that work with a latent space.

Table 5.1: Example of nominal and ordinal output coding for five classes ($Q = 5$).

1-of-Q coding	*OrderedPartitions*
$\begin{pmatrix} +1 & -1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 & -1 \\ -1 & -1 & +1 & -1 & -1 \\ -1 & -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & -1 & +1 \end{pmatrix}$	$\begin{pmatrix} +1,-1,-1,-1,-1 \\ +1,+1,-1,-1,-1 \\ +1,+1,+1,-1,-1 \\ +1,+1,+1,+1,-1 \\ +1,+1,+1,+1,+1 \end{pmatrix}$

decoding approach [126] is applied, i.e. the chosen label is that which minimizes the exponential loss:

$$\hat{y} = \arg \min_{1 \leq q \leq Q} d_{\mathrm{L}} \left(\mathbf{M}_q, \mathbf{g}(\mathbf{x}) \right),$$

where $\hat{y}$ is the predicted class label, being $\hat{y} \in \mathcal{Y} = \{C_1, C_2, \ldots, C_Q\}$ containing Q labels, $\mathbf{M}_q$ is the code associated to class C_q (i.e. each of the rows of coding at the right of Table 5.1), $\mathbf{g}(\mathbf{x}) = f(\mathbf{x}, \theta)$ is the vector of predictions given by the MLP model in Eq. (4.9) (see Section 4.5.2 in Chapter 4), and $d_{\mathrm{L}} \left(\mathbf{M}_q, \mathbf{g}(\mathbf{x}) \right)$ is the exponential loss function:

$$d_{\mathrm{L}} \left(\mathbf{M}_q, \mathbf{g}(\mathbf{x}) \right) = \sum_{i=1}^{Q} \exp \left(-\mathbf{M}_{iq} \cdot g_i(\mathbf{x}) \right). \tag{5.1}$$

5.2.2 *Proposed method*

This section presents our Evolutionary Extreme Learning Machine for Ordinal Regression (E-ELMOR) model and the associated training algorithm. First, the E-ELMOR extends the ELMOR model to obtain a probabilistic output. For doing that, the softmax transformation layer [199] is added to the ELMOR model using the negative exponential losses of Eq. (5.1):

$$p_q = p_q(\mathbf{x}, \boldsymbol{\theta}_q) = \frac{\exp(-d_{\mathrm{L}} \left(\mathbf{M}_q, \mathbf{g}(\mathbf{x}) \right))}{\sum_{i=1}^{Q} \exp(-d_{\mathrm{L}} \left(\mathbf{M}_i, \mathbf{g}(\mathbf{x}) \right))}, 1 \leq q \leq Q, \tag{5.2}$$

where p_q is the posterior probability that a pattern $\mathbf{x}$ has of belonging to class C_q and this probability should be maximized for the actual class and minimized (or ideally be zero) for the rest of the classes. This formulation is used for evaluating the individuals in the evolutionary process but not for solving the ELMOR system of equations.

In the case of ordinal regression, the posterior probability must decrease from the true class to more distant classes. This has been pointed out in the work of Pinto da Costa et al. [18]. In that work an unimodal output function is imposed to the neural network model, and the probability function monotonically decreases as the classes are more distant from the true one.

Table 5.2: Example of an absolute cost matrix ($\mathbf{A}$) and an absolute cost matrix plus the matrix of ones ($\mathbf{C} = \mathbf{A} + \mathbf{1}$) for five classes ($Q = 5$).

$\mathbf{A}$	$\mathbf{C} = \mathbf{A} + \mathbf{1}$
$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 2 & 3 \\ 4 & 3 & 2 & 1 & 2 \\ 5 & 4 & 3 & 2 & 1 \end{pmatrix}$

According to the previous observation, we propose a fitness function for guiding the evolutionary optimization that simultaneously considers two features of a classifier:

1. Misclassification of non-adjacent classes should be more penalized as the difference between classes' labels grows.

2. The posterior probability should be unimodal and monotonically decrease for non-adjacent classes.

In this way, not only the right class output is considered, but also the posterior probabilities with respect to the wrong classes are reduced. In order to satisfy these restrictions, we propose the weighted root mean square error ($WRMSE$).

First, we design the type of cost associated with the errors. Let us define the absolute cost matrix as $\mathbf{A}$, where the element $a_{ij} = |i - j|$ is equal to the difference in the number of categories, $a_{ij} = |i - j|$. The absolute cost matrix is used, for instance, for calculating the MAE, being i the actual label and j the predicted label. An example of an absolute cost matrix for five classes is shown in Table 5.2. In the case of $WRMSE$, $\mathbf{A}$ cannot be directly applied because it would suppress information about the posterior probability of the correct class (see Eq. (5.3)). Then, we add a square matrix of ones $\mathbf{1}$ so that our final cost matrix is $\mathbf{C} = \mathbf{A} + \mathbf{1}$ (see an example in Table 5.2).

Second, according to the model output defined in Eq. (5.2), we define the weighted root mean square error ($WRMSE$) associated to a pattern as:

$$e = \frac{\sum_{q=1}^{Q} (c_{iq} \sqrt{(t_q - p_q)^2})}{Q}, \tag{5.3}$$

where i is index of the true target and c_{iq} represents the cost of errors associated to the q output t_q of the neural network coded in matrix $\mathbf{C}$ (see Table 5.2). Finally, the total error of the prediction is defined as:

$$WRMSE = \frac{\sum_{i=1}^{N} (e_i)}{N}. \tag{5.4}$$

For ending this section, it should be noticed that in a single model multi-class classifier the $RMSE$ has the interesting property of selecting solutions that consider

Table 5.3: Characteristics of the benchmark datasets

Dataset	#Pat.	#Attr.	#Classes	Class distribution
automobile (AU)	205	71	6	(3, 22, 67, 54, 32, 27)
balance-scale (BS)	625	4	3	(288, 49, 288)
bondrate (BO)	57	37	5	(6, 33, 12, 5, 1)
contact-lenses (CL)	24	6	3	(15, 5, 4)
eucalyptus (EU)	736	91	5	(180, 107, 130, 214, 105)
LEV (LE)	1000	4	5	(93, 280, 403, 197, 27)
newthyroid (NT)	215	5	3	(30, 150, 35)
pasture (PA)	36	25	3	(12, 12, 12)

good classification performance of all classes simultaneously [108]. In the case of MZE, only one network output (the one with maximum value) contributes to the error function, and it does not contribute with the output's value. However, for $RMSE$ it is straightforward to check that each model's output (posterior probabilities) contributes to the error function. Then, the model's decision thresholds and posteriors will tend to be more discriminative. This implicit pressure over the posteriors is even more severe in the case of $WRMSE$.

5.2.3 *Experimental section*

This section presents experiments comparing the present approach with several alternatives, with special attention to the E-ELM and the ELMOR as reference methods.

5.2.3.1 *Datasets and related methods*

Table 5.3 shows the characteristics of the eight datasets included in the experiments. The table shows the number of patterns, attributes and classes, and also the class distribution (number of patterns per class). The publicly available real ordinal regression datasets were extracted from benchmark repositories (UCI [231] and `mldata.org` [244]). The experimental design includes 30 stratified random splits (with 75% of patterns for training and the remainder for generalization).

In addition to the E-ELM, ELMOR and the proposed method (E-ELMOR), we include the following alternatives in the experimental section:

- The POM algorithm [138], with the *logit* link function.
- The GPOR method [41].
- NNOR [130] neural network with decomposition scheme by Frank and Hall in [27].

The algorithms' hyperparameters were adjusted by a grid search using MAE as parameter selection criteria. For NNOR, the number of hidden neurons, M, was selected

by considering the following values, $M \in \{5, 10, 20, 30, 40\}$. The sigmoidal activation function was considered for the hidden neurons. For ELMOR, E-ELM and E-ELMOR, higher numbers of hidden neurons are considered, $M \in \{5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$, given that it relies on sufficiently informative random projections [131]. With regards to the GPOR algorithm, the hyperparameters are determined by part of the optimization process. For E-ELM and E-ELMOR the evolutionary parameters' values are the same as used at [176]. The number of iterations was 50 and the population size 40.

Table 5.4: Comparison of E-ELMOR to other nominal and ordinal related classification methods. The mean and standard deviation of the generalization results are reported for each dataset, as well as the mean ranking. The best result is in bold face and the second best result in italics.

	MZE Mean								Mean MZE rank
Method/DataSet	AU	BS	BO	CL	EU	LE	NT	PA	
E-ELM	0.453	0.152	0.544	0.344	0.507	0.393	0.152	0.389	4.94
ELMOR	0.384	0.082	*0.476*	0.383	0.440	**0.371**	0.051	0.389	3.31
GPOR	0.389	**0.034**	**0.422**	0.394	**0.315**	0.388	*0.034*	0.478	3.13
NNOR	*0.376*	*0.039*	0.500	**0.294**	0.418	0.373	0.035	**0.237**	**2.31**
POM	0.533	0.092	0.656	0.378	0.841	0.380	**0.028**	0.504	4.69
E-ELMOR	**0.360**	0.092	0.533	*0.306*	*0.394*	*0.372*	0.035	*0.333*	*2.63*
	MAE Mean								Mean MAE rank
Method/DataSet	AU	BS	BO	CL	EU	LE	NT	PA	
E-ELM	0.688	0.216	0.722	0.517	0.718	0.439	0.154	0.404	5.06
ELMOR	0.542	0.089	0.649	0.522	0.531	**0.406**	0.052	0.404	3.44
GPOR	0.594	**0.034**	**0.624**	0.511	**0.331**	0.422	*0.034*	0.489	2.75
NNOR	**0.503**	*0.044*	0.671	*0.456*	0.476	0.408	0.035	**0.241**	*2.44*
POM	0.953	0.111	0.947	0.533	2.029	0.415	**0.028**	0.585	5.00
E-ELMOR	*0.510*	0.108	*0.644*	**0.433**	*0.447*	*0.407*	0.035	*0.344*	**2.31**
	AMAE Mean								Mean AMAE rank
Method/DataSet	AU	BS	BO	CL	EU	LE	NT	PA	
E-ELM	0.813	0.426	1.119	0.545	0.778	0.632	0.212	0.404	4.75
ELMOR	0.649	0.176	1.168	0.531	0.575	0.611	0.114	0.404	3.94
GPOR	0.792	**0.051**	1.360	0.651	**0.362**	0.654	0.062	0.489	4.13
NNOR	**0.566**	*0.066*	1.135	*0.493*	0.506	**0.608**	0.059	**0.241**	*2.19*
POM	1.026	0.107	*1.103*	0.535	1.990	0.632	**0.050**	0.585	4.06
E-ELMOR	*0.592*	0.172	**1.041**	**0.463**	*0.489*	**0.608**	*0.052*	*0.344*	**1.94**

5.2.3.2 *Experimental results*

Table 5.4 shows mean generalization performance of all the algorithms including metrics described at Section 3.2.4. The mean rankings of MZE, MAE and $AMAE$ are obtained to compare the different methods. A Friedman's non-parametric test

Table 5.5: Table with the different algorithms compared with E-ELMOR using the Holm procedure ($\alpha = 0.10$) in terms of $AMAE$. The horizontal line shows the division between methods significantly different from E-ELMOR.

i	Algorithm	z	p	α'_{Holm}
1	E-ELM	3.0067	0.0026	0.0200
2	GPOR	2.3385	0.0194	0.0250
3	POM	2.2717	0.0231	0.0333
4	ELMOR	2.1381	0.0325	0.0500
5	NNOR	0.2673	0.7893	0.1000

for a significance level of $\alpha = 0.05$ has been carried out to determine the statistical significance of the differences in rank in each method. The test rejected the null-hypothesis stating that all algorithms performed equally in the mean ranking of the three metrics. Because of space restrictions, we will only examine $AMAE$ metric, since it is the most robust one. For this purpose, we have applied the Holm post-hoc test to compare E-ELMOR to all the other classifiers in order to justify our proposal. The Holm test is a multiple comparison procedure that work with a control algorithm (E-ELMOR) and compares it to the remaining methods [240]. Results of the test are show in Table 5.5, which shows that our proposal improves on all the methods' performance except NNOR for $\alpha = 0.10$, and there are only statistical differences with E-ELM for $\alpha = 0.05$. The second best performance in $AMAE$ was for NNOR.

5.2.4 *Conclusions and future work*

In this section, we have adapted the ELMOR model to the Evolutionary ELM. We have proposed the weighed Root Mean Squared Error (RMSE) error function to guide the algorithm. Based on theoretical analysis and experimental results, we justify the proposal compared to the reference methods and other ordinal regression techniques. Future work involves the design and experiments with new output codes and associated error functions.

5.3 PROPOSAL 2: LATENT VARIABLE MODELLING WITH PROBABILITY DISTRIBUTIONS

In this section we present the second proposal for explicitly dealing with ordinal classification problems. The approach lies between the regular regression models and the threshold models. The regular regression is addressed by generating random values from triangular probability distributions. There is a different triangular distribution for modelling (generating) the random values of each ordinal class. For each pattern, these values are considered as the regression response variable. In this way we turn a classification problem into a regression problem. Then, the methodology is suitable for any kind of regression model. With this problem reformulation

we are imposing the class order in the *new* regression dataset, and we assume that the trained regressor should reflect this order.

Regarding the threshold model, the limits of the triangular distributions are used as thresholds for properly assigning a pattern to a class. Even thought the simplicity of this idea, experimental results show that the proposal is competitive regarding state-of-the-art ordinal classification methods, specially when considering an order sensitive performance metric.

5.3.1 *Numerical Variable Reconstruction*

The center of this first proposal is to turn a classification problem into a regression problem so that the class structures are reflected in the regression variable. This procedure is called Numerical Variable Reconstruction (NVR). NVR has two main steps: the numerical variable reconstruction itself, i.e. the random values generation during the classifier training procedure; and the classification procedure based on the predicted values of the regression response variable.

5.3.1.1 *Theoretical basis*

Let us consider an ordinal classification problem as defined in Section 3.2. Our goal is to find a classifier g that is capable of assigning, according to the best fit possible, a pattern to its class depending on its characteristics. It should also be designed to include the information related to the ordinality of the classes. Thus, the ordered structure of $\mathcal{Y}$ should be used to determine g.

As previously mentioned, a simple and intuitive way to make this calculation is to assume the existence of a one-dimensional latent variable $z = \phi(\mathbf{x})$ that is a function of the characteristics observed and takes on the underlying order mentioned in the previous paragraph. Note that for convenience, in this section we express the latent variable formulation in a different way of the formulation in Section 3.3.3. The concept of order is included in that the lower values of z will most likely correspond to class C_1, reaching a limit, say z_1, on class C_2 and so on for the rest of the classes[2]. Thus, the classifier will be given by

$$g(\mathbf{x}) = C_q \text{ if } \phi(\mathbf{x}) \in (z_{q-1}, z_q),$$

where we allow $(z_0, z_Q) = (-\infty, \infty)$.

This is one of the approaches to this problem, obtaining different solutions depending on the kind of function considered for ϕ and the strategy used to determine the limit values for z. The drawback here is how to choose an optimal projection function, such that the limits can be adequately adjusted comparing the observed classifications and estimated over the training set $\mathbf{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^K, y_i \in \mathcal{Y}, i = 1, \ldots, N\}$, with $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iK})$ (see ordinal regression problem formulation in Section 3.2).

Our approach does not attempt to determine z (and therefore ϕ). Instead, we present it as a random variable according to the distribution function $u = F(z) =$

2 Note in other sections we use the symbol θ for these thresholds. Here we use z and u in order to difference between the latent unobservable variable z and the random observable variable u.

$G(\mathbf{x}) = F(\phi(\mathbf{x}))$, where due to the inherent monotony in F, we can have some advantages with respect to the previously mentioned procedure:

- $F(z)$ is a uniform random variable in the interval $[0,1]$, regardless of how z is a function of $\mathbf{x}$. Thus, not knowing ϕ and F is no longer inconvenient.
- Since the values for z_q determine the limits between classes, and the distribution function is monotonous, its images $u_q = F(z_q)$ will restrain the probabilities to the interval that belongs to each class. Also, since the relative size f_i of those intervals in $\mathbf{D}$ estimates its probability, the boundaries will be given by

$$0 = u_0 < \hat{u}_1 = f_1 < \hat{u}_2 = f_1 + f_2 < \ldots < u_Q = 1.$$

Provided all this, each pattern in $\mathbf{D}$ is assigned to a random number. In order to do this, and assuming we are distributing pattern i that belongs to class C_q, we generate a value in the interval $(\hat{u}_{q-1}, \hat{u}_q)$ using the F_q probability distribution, which generates values in that interval. Note there is a different F_q modelling each class C_q. Calling this value U_i and bearing in mind the relationship between u and $\mathbf{x}$ we have a new set $\mathbf{D}' = \{(\mathbf{x}_i, U_i) : \mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^k,\ U_i \in [0,1]\}$ where the new response values meet two conditions:

- They indicate the class they belong to, given the way they have been assigned.
- They can be obtained theoretically using $U = G(\mathbf{x})$.

Thus, the initial classification problem becomes a regression one, expected to be easier to solve in order to estimate the function G.

Once the value for $\hat{G}$ is obtained (estimated from G), the classification of new patterns becomes immediate assigning class q as long as we can verify $\hat{u}_{q-1} < \hat{G}(\mathbf{x}) < \hat{u}_q$.

The training procedure, then, can be stated as follows:

1. Calculate, using the training set $\mathbf{D}$, the a priori probabilities of each one of the classes: $f_q = \frac{n_q}{N}$ where n_q and N are the sizes with respect to class C_q and the set $\mathbf{D}$.
2. Obtain the classification limits using the empirical distribution function $\hat{u}_q = \sum_{j=0}^{q} f_j$.
3. For each n_q patterns belonging to class C_q, generate a random value U_i using the triangular symmetric distributions in the interval between $\hat{u}_{q-1}$ and $\hat{u}_q$.
4. Make a regression model considering the sample $(\mathbf{x}, U)$ where vectors $\mathbf{x}$ of each class are paired with the U values of the previous step. Let $\hat{G}$ be the resulting regressor, which is independent of the model and the training algorithm.

Finally, for every unseen pattern $\mathbf{x}$, $\hat{G}(\mathbf{x})$ is determined and assigned to class C_q, verifying $\hat{u}_{q-1} < \hat{G}(\mathbf{x}) < \hat{u}_q$.

5.3.1.2 *NVR using the triangular probability distribution*

In order to simplify the mathematical issues regarding the underlying probability distribution for $\mathcal{U}$ modelling, we choose the triangular distribution for this model. The triangular distribution is a continuous probability distribution with lower limit a, upper limit b and mode c, where $a < b$ and $a \leq c \leq b$. Since the triangular distribution is continuous, the inverse transform sampling method can be used to generate random values from the uniform distribution [245], which is the common distribution provided by every programming language or environment.

For a Q class problem, $U_q, q = 1, 2, \ldots, Q$, different random variables are used to generate values representing all the patterns $\mathbf{x} \in C_q$. Therefore, Q triangular distributions can be used for modelling these U_q random variables which compose $\mathcal{U}$. Thus, each triangular distribution will be defined as $F_q(U_q|a_q, b_q, c_q)$.

Every triangular distribution must be adapted based on the assumptions done in the previous section. Firstly, the a_q and b_q parameters must be adjusted so that the $b_q - a_q$ distance is proportional to the a priori probability n_q/N a pattern has of belonging to a class C_q. Regarding parameter c_q, for simplicity, it is defined as the middle point between a_q and b_q, although in general it can be any value in the interval $[a_q, b_q]$. In our approach, all the triangular distributions are adjusted under the restriction of $\mathcal{U} \in [0, 1]$

Once the triangular distributions are set up, they can be used for generating random values z_i for each pattern $\mathbf{x}_i$ by using the corresponding $F_q(U_q|a_q, b_q, c_q)$, therefore:

$$z_i = F_q(U_q|a_q, b_q, c_q) \text{ for } \mathbf{x}_i \in C_q. \tag{5.5}$$

At the end of this procedure the ordinal classification problem will be turned into a regression problem in the form of $\hat{G} = \phi(\mathbf{x})$, where ϕ is a regression model.

Assuming we have a trained regression model ϕ for predicting $\hat{G}$ values, these estimated values must be mapped to classes in order to perform the original classification task. In the case of the triangular distribution, this is straightforward. That is because the Q different thresholds u_q, described in Subsection 5.3.1.1, correspond to the a_{q+1} and b_q parameters. Then, for classifying a pattern in class C_1, $0 \leq \hat{G} \leq u_1$, where $u_1 = b_1 = a_2$. A pattern will be assigned to class C_2, if $u_1 < \hat{G} \leq u_2$, where $u_2 = b_2 = a_3$, and so on. Regarding the last class C_Q, patterns will be classified in this class if $u_{Q-1} < \hat{G} \leq 1$, where $u_{Q-1} = b_{Q-1}$. An example of the triangular distributions parameters set up can be seen in Fig. 5.1.

5.3.2 *Preliminary Experiments*

This section presents the experiments done for evaluating the NVR proposal.

5.3.2.1 *Ordinal classification datasets and experimental design*

For these experiments, we have collected a set of real ordinal classification datasets which are publicly available at the UCI repository [231] and at the *mldata.org* datasets repository [246] (see Table 5.6 for datasets description).

Regarding the experimental design, we have considered two different options depending on the stochastic nature of the method. The proposed method (NVR) is

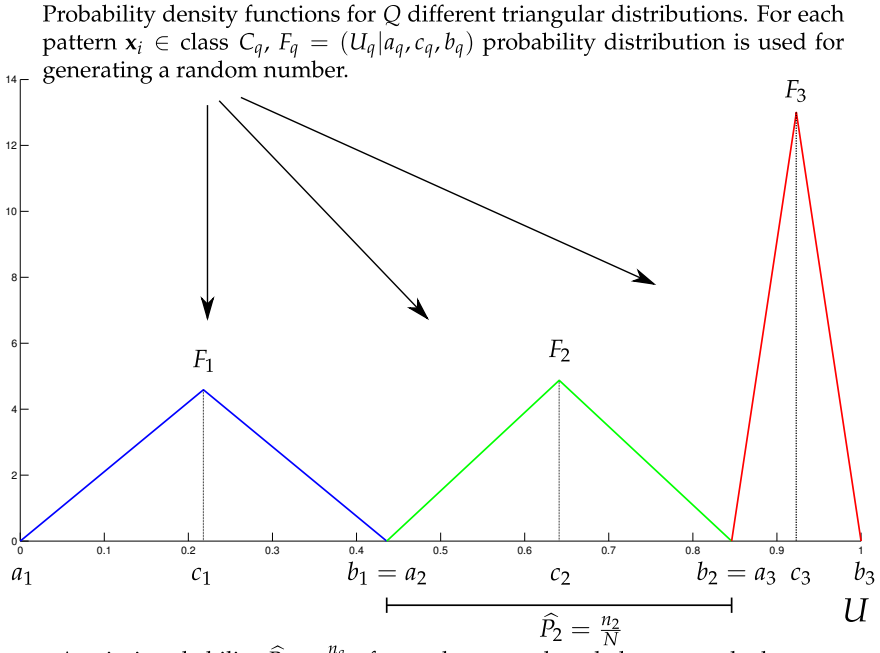


Figure 5.1: NVR with triangular probability distributions example.

Table 5.6: Datasets used for the experiments

Dataset	Size	#Input	#Classes	Classes Distribution
automobile	205	71	6	(3,22,67,54,32,27)
balance-scale	625	4	3	(288,49,288)
ERA	1000	4	9	(92,142,181,172,158,118,88,31,18)
LEV	1000	4	5	(93,280,403,197,27)
tae	151	54	3	(49,50,52)

non-deterministic because the U_i values are randomly generated using the triangular distributions and the seed used for the method determines the obtained results. For this method, we perform 10 times a holdout validation and 3 repetitions for each holdout (obtaining a total of $10 \times 3 = 30$ different results). Each holdout is a stratified random division of the data, where approximately 75% of the instances are used for the training set and 25% of them for the test set (maintaining the original distribution of classes for both sets). For the deterministic methods (all of them except NVR), we perform 30 times a stratified holdout validation using 75% of the instances for the training set and 25% of them for the generalization set, what implies a total of 30 different results. The partitions are the same for all the deterministic methods.

In this way, a total of 30 error measures has been obtained for all the methods compared, which guarantees a proper statistical significance of the results.

5.3.2.2 *Experimental results*

Table 5.7: Comparison of SVR-NVR to other ordinal classification methods and SVC: Results of MZE ($MZE_G(\%)$) and MAE (MAE_G) on the generalization set

Dataset	Method	MZE mean	MZE std	MAE mean	MAE std
automobile	SVC	*0.3423*	*0.0659*	0.4205	0.0837
	GPOR-ARD	0.3891	0.0726	0.5942	0.1307
	RED-SVM	**0.3160**	**0.0548**	**0.3929**	**0.0730**
	SVOREX	0.6333	0.0685	0.9276	0.1149
	SVORIM	0.6385	0.0552	0.9327	0.0918
	SVR-NVR	0.3449	0.0746	*0.4128*	*0.1052*
balance-scale	SVC	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	GPOR-ARD	0.0342	0.0118	0.0342	0.0118
	RED-SVM	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	SVOREX	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	SVORIM	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	SVR-NVR	0.0571	0.0213	0.0597	0.0229
ERA	SVC	0.7492	0.0194	1.2575	0.0744
	GPOR-ARD	**0.7121**	**0.0270**	1.2413	0.0505
	RED-SVM	0.7551	0.0239	1.2172	0.0431
	SVOREX	*0.7349*	*0.0305*	*1.2108*	*0.0363*
	SVORIM	0.7579	0.0213	1.2143	0.0343
	SVR-NVR	0.7409	0.0182	**1.1996**	**0.0528**
LEV	SVC	**0.3723**	**0.0297**	0.4080	0.0343
	GPOR-ARD	0.3877	0.0301	0.4219	0.0308
	RED-SVM	0.3764	0.0238	0.4133	0.0265
	SVOREX	*0.3724*	*0.0218*	*0.4073*	*0.0244*
	SVORIM	0.3757	0.0289	0.4119	0.0318
	SVR-NVR	0.3744	0.0304	**0.4007**	**0.0308**
tae	SVC	0.5281	0.0896	0.5886	0.0834
	GPOR-ARD	0.6719	0.0407	0.8614	0.1551
	RED-SVM	0.4781	0.0735	0.5149	0.0865
	SVOREX	*0.4421*	*0.0752*	*0.4816*	*0.0922*
	SVORIM	0.4711	0.0836	0.5149	0.0851
	SVR-NVR	**0.4044**	**0.0305**	**0.4360**	**0.0256**

For comparison purposes, different state-of-the-art methods have been included in the experimentation (for more details about these methods, please check Section 5.4.7.2):

- Gaussian Processes for Ordinal Regression (GPOR) [41] with ARD feature selection (we will refer the algorithm to as GPOR-ARD).

- Support vector ordinal regression with implicit (SVORIM) and explicit (SVORIM) constraints [33, 39].
- Reduction from cost-sensitive ordinal ranking to weighted binary classification (RED) with SVM (RED-SVM) [30, 31].
- The Cost Support Vector Classification (SVC) available in libSVM 3.0 [169, 239] is included as a reference nominal classifier.

In our approach, the SVR algorithm is used as the regressor model, so the method is called SVR-NVR. The ϵ-SVR available in libSVM [169, 239] is used.

Regarding the algorithms' hyper-parameters, the following procedure has been applied. For the Support Vector algorithms, i.e. SVC, RED-SVM, SVOREX, SVORIM and ϵ-SVR, the corresponding hyper-parameters (regularization parameter, C, and width of the Gaussian functions, γ), were adjusted using a grid search with a 10-fold cross-validation, considering the following ranges: $C \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$ and $\gamma \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$. Regarding ϵ-SVR, it has the additional ϵ parameter. The ϵ parameter values were $\epsilon \in \{10^{0}, 10^{1}, \ldots, 10^{3}\}$. For GPOR-ARD no hyper-parameters were set up since the method optimizes the associated parameters itself. All the methods were configured to use the Gaussian kernel.

Table 5.8: Average MZE and MAE results and mean rankings

	SVR-NVR	SVC	GPOR-ARD	RED-SVM	SVOREX	SVORIM
$\overline{MZE}_G$	**0.3843**	0.4390	0.3984	*0.3851*	0.4365	0.4486
$\overline{R}_{MZE_G}$	3.2	4.4	*2.9*	3.5	**2.7**	4.3
$\overline{MAE}_G$	**0.5018**	0.6306	0.5349	*0.5077*	0.6055	0.6148
$\overline{R}_{MAE_G}$	**2.2**	5.2	3.9	3.2	*2.7*	3.8

The experiments have been carried out by following the experimental design described in previous subsection. Results considering mean and standard deviation in MZE and MAE are showed in Table 5.7 (see Section 3.2.4 in Chapter 3 for metrics description). The best statistical result is in bold face and the second best result in italics.

In order to compare these results, a non-parametric Friedman [247] test has been employed. The average MZE_G and average MAE_G results in the generalization sets were used for the Friedman test. The test accepted the null-hypothesis that all algorithms perform equally well when $\alpha = 0.05$. Therefore, we can conclude that our method reaches the state-of-the-art methods regarding ordinal classification. In addition, for information purposes, Table 5.8 shows the average results in MZE_G and MAE_G for each method, as well as the average ranking for each performance metric, this is $\overline{R}_{MZE_G}$ and $\overline{R}_{MAE_G}$. Table 5.8 shows that SVR-NVR has the best average MZE results and the third mean MZE rank, whereas it has the best performance for average MAE results and mean ranking.

5.3.3 *Conclusions*

In this section, a novel method for generally adapting classification and regression models, such as artificial neural networks or support vector machines, was presented. The ordinal classification problem is reformulated as a regression problem by the reconstruction of a numerical variable, which represents the different ordered class labels.

The NVR algorithm is implemented with the triangular probability distribution for the variable reconstruction, and with the Support Vector Regression algorithm as the regression method. Experimental results demonstrate that our method reaches the state-of-the-art related algorithms. Despite the simplicity and generality of the method, results are competitive in comparison with very specific methods for ordinal regression.

5.4 PROPOSAL 3: PAIRWISE CLASS DISTANCES PROJECTION FOR ORDINAL CLASSIFICATION

In this section the PCD and PCDOC proposal is fully described and the PCD projection is analysed with some synthetic datasets in order to observe its suitability both for linear and nonlinear cases. Then, the section follows with controlled experiments to observe the influence of dimensionality, class overlapping and data multi-modality in the performance of PCDOC. The next part presents a thorough analysis of results with benchmark datasets from real world problems. Finally, the last part sums up key conclusions and points to future work.

5.4.1 *Introduction and motivation*

While threshold approaches offer an interesting perspective on the problem of ordinal classification, they learn the projection from the input space onto the one-dimensional latent space only indirectly as part of the overall model fitting. As with any latent model fitting, direct construction hints one may have about the desired form of the latent model can prove very useful for obtaining high quality models. The key idea of the PCDOC is to construct such a projection model directly, using insights about the class distribution obtained from pairwise distance calculations. Indeed, our motivation stems from the fact that the order information should also be present in the data input space and it could be interesting to take advantage from it to construct an useful variable for ordering the patterns using the ordinal scale. Additionally, regression is clearly the most natural way to approximate this continuous variable. As a result, we propose to construct the ordinal classifier in two stages: 1) the input data is first projected into a one dimensional variable by considering the relative position of the patterns in the input space, and, 2) a standard regression algorithm is applied to learn a function to predict new values of this derived variable.

The main contribution of the PCDOC is the projection onto a one dimensional variable, which is done by a guided projection process (PCD). This process exploits the ordinal space distribution of patterns in the input space. A measure of how *'well'* a pattern is located within its corresponding class region is defined by considering

the distances between patterns of the adjacent classes in the ordinal scale. Then, a projection interval is defined for each class, and the centres of those intervals (for non-boundary classes) are associated with the *'best'* located patterns for the corresponding classes (quantified by the measure mentioned above). For the boundary classes (first and last in the class order), the extreme end points of their projection intervals are associated with the most separated patterns of those classes. All the other patterns are assigned proportional positions in their corresponding class intervals, again according to their *'goodness'* values expressing how *'well'* a pattern is located within its class. We refer to this projection as *Pairwise Class Distance* (PCD) based projection. In Section 5.4.4, the behaviour of this projection is evaluated over synthetic datasets, showing an intuitive response and a good ability to separate adjacent classes even in nonlinear settings.

Once the mapping is done, our framework allows to design effective ordinal algorithms based on well-tuned regression approaches. The final classifier constructed by combining PCD and a regressor is called *Pairwise Class Distances for Ordinal Classification* (PCDOC). In this contribution, PCDOC is implemented using ϵ-Support Vector Regression (ϵ-SVR) algorithm [44, 248] as the base regressor, although any other properly handled regression method could be used. We carry out an extensive set of experiments on ten real world ordinal regression datasets, comparing our approach with eight state-of-the-art methods. Our method, though simple, holds out very well. Under four complementary performance metrics, the proposed method obtained the best mean ranking for three of the four metrics.

5.4.2 *Latent variable modelling formulation*

Latent variable models or threshold consider the ordinal scale as the result of coarse measurements of a continuous variable, called the latent variable. The threshold model can be represented with the following general expression:

$$f(\mathbf{x}, \boldsymbol{\theta}) = \begin{cases} C_1, & \text{if } g(\mathbf{x}) \leq \theta_1, \\ C_2, & \text{if } \theta_1 < g(\mathbf{x}) \leq \theta_2, \\ \vdots & \\ C_Q, & \text{if } g(\mathbf{x}) > \theta_{Q-1}, \end{cases} \tag{5.6}$$

where $g : \mathcal{X} \to \mathbb{R}$ is the function that projects data space onto the one-dimensional latent space $\mathcal{Z} \subseteq \mathbb{R}$ and $\boldsymbol{\theta} = \theta_1, \ldots, \theta_{Q-1}$ (where $\theta_1 < \ldots < \theta_{Q-1}$) are the thresholds that divide the space into ordered intervals corresponding to the classes.

In the PCDOC proposal, it is assumed that a model $\phi : \mathcal{X} \to \mathcal{Z}$ can be found that links data items $\mathbf{x} \in \mathcal{X}$ with their latent space representation $\phi(\mathbf{x}) \in \mathcal{Z}$. We place that proposal in the context of latent variable models for ordinal classification because of its similarity to these models. In contrast to other models employing a one dimensional latent space, e.g. POM [36], we do not consider variable thresholds, but impose fixed values for $\boldsymbol{\theta}$. However, suitable dimensionality reduction is given due attention: first, by trying to exploit the ordinal structure of the space $\mathcal{X}$, and second we explicitly put external pressure on the margins between the classes in $\mathcal{Z}$ (see Section 5.4.4).

Our approach is different from the previous ones in that it does not implicitly learn latent representations of the training inputs. Instead, we impose how training inputs $\mathbf{x}_i$ are going to be represented through $z_i = \phi(\mathbf{x}_i)$. Then, this representation is generalized to the whole input space by training a regressor g on the $(\mathbf{x}_i, z_i)$ pairs, resulting in a projection function $g : \mathcal{X} \rightarrow \mathcal{Z}$. To ease the presentation, we will sometimes write training input patterns $\mathbf{x}$ as $\mathbf{x}^{(q)}$ to explicitly reflect their class label rank q (i.e. the class label of $\mathbf{x}$ is C_q).

5.4.3 *Pairwise Class Distance (PCD) projection*

To describe the Pairwise Class Distance (PCD) projection, first, we define a measure $w_{\mathbf{x}^{(q)}}$ of *"how well"* a pattern $\mathbf{x}^{(q)}$ is placed within other instances of class C_q, by considering its Euclidean distances to the patterns in adjacent classes. This is done on the assumption of ordinal pattern distribution in the input space $\mathcal{X}$. For calculating this measure, the minimum distances of a pattern $\mathbf{x}_i^{(q)}$ to patterns in the previous and next classes, C_{q-1} and C_{q+1}, respectively, are used. The minimum distance to the previous/next class is

$$\kappa(\mathbf{x}_i^{(q)}, q \pm 1) = \min_{\mathbf{x}_j^{(q\pm1)}} \left\{ ||\mathbf{x}_i^{(q)} - \mathbf{x}_j^{(q\pm1)}|| \right\}, \tag{5.7}$$

where $||\mathbf{x} - \mathbf{x}'||$ is the Euclidean distance between $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^K$. Then,

$$w_{\mathbf{x}_i^{(q)}} = \begin{cases} \dfrac{\kappa(\mathbf{x}_i^{(q)}, q+1)}{\max_{\mathbf{x}_n^{(q)}} \left\{ \kappa(\mathbf{x}_n^{(q)}, q+1) \right\}}, & \text{if } q = 1, \\ \dfrac{\kappa(\mathbf{x}_i^{(q)}, q-1) + \kappa(\mathbf{x}_i^{(q)}, q+1)}{\max_{\mathbf{x}_n^{(q)}} \left\{ \kappa(\mathbf{x}_n^{(q)}, q-1) + \kappa(\mathbf{x}_n^{(q)}, q+1) \right\}}, & \text{if } q \in \{2, \ldots, Q-1\}, \\ \dfrac{\kappa(\mathbf{x}_i^{(q)}, q-1)}{\max_{\mathbf{x}_n^{(q)}} \left\{ \kappa(\mathbf{x}_n^{(q)}, q-1) \right\}}, & \text{if } q = Q, \end{cases} \tag{5.8}$$

where the sum of the minimum distances of a pattern with respect to adjacent classes is normalized across all patterns of the class, so that $w_{\mathbf{x}_i^{(q)}}$ has a maximum value of 1.

Figure 5.2 shows the idea of minimum distances for each pattern with respect to the patterns of the adjacent classes. In this figure, patterns of the second class are considered. The example illustrates how the $w_{\mathbf{x}^{(2)}}$ value is obtained for the pattern $\mathbf{x}^{(2)}$ marked with a circle. For distances between $\mathbf{x}^{(2)}$ and class 1 patterns, the item $\mathbf{x}^{(1)}$ has the minimum distance, so $\kappa(\mathbf{x}^{(2)}, 1)$ is calculated by using this pattern. For distances between $\mathbf{x}^{(2)}$ and class 3 patterns, $\kappa(\mathbf{x}^{(2)}, 3)$ is the minimum distance between $\mathbf{x}^{(2)}$ and $\mathbf{x}^{(3)}$.

By using $w_{\mathbf{x}_i^{(q)}}$, we can derive a latent variable value $z_i \in \mathcal{Z}$. Before continuing, thresholds must be defined in order to stablish the intervals on $\mathcal{Z}$ which correspond to each class, so that calculated values for z_i may be positioned on the proper interval. Also, predicted values $\hat{z}_i$ of unseen data would be classified in different classes according to these thresholds (see Subsection 5.4.5), in a similar way to any other

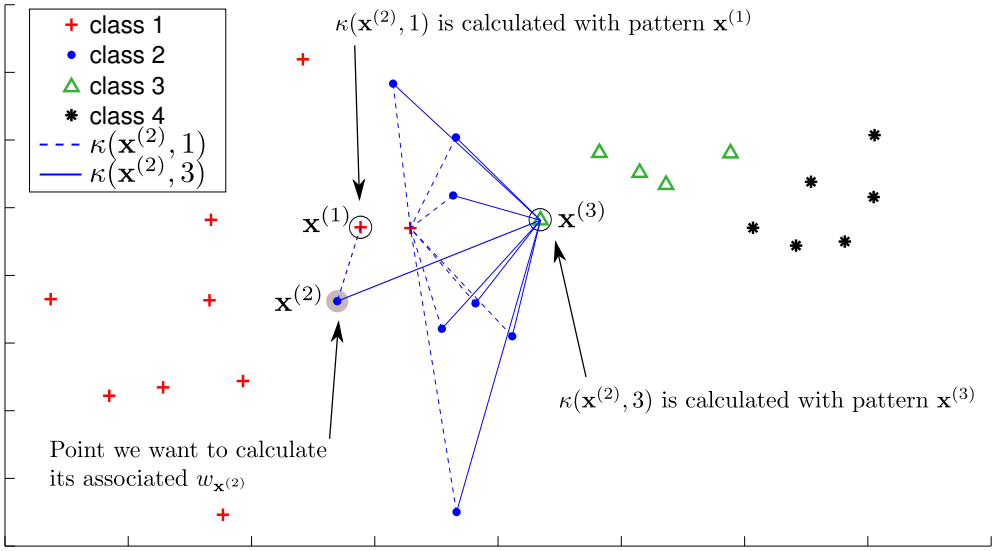


Figure 5.2: Illustration of the idea of minimum Pairwise Class Distances. All the minimum distances of patterns of class C_2 regarding patterns of adjacent classes are painted with lines. $\mathbf{x}^{(2)}$ is the point we want to calculate its associated $w_{\mathbf{x}^{(2)}}$.

threshold model. For the sake of simplicity, $\mathcal{Z}$ is defined between 0 and 1, and the thresholds are positioned in the uniform manner[3] :

$$\theta = \{\theta_1, \theta_2, \ldots, \theta_Q\} = \{1/Q, 2/Q, \ldots, 1\}. \tag{5.9}$$

Considering θ, the centres $c_q \in \{c_1, c_2, \ldots, c_Q\}$ for $\mathcal{Z}$ values belonging to class C_q are set to: $c_1 = 0$, $c_Q = 1$ and

$$c_q = \frac{q}{Q} - \frac{1}{2Q}, \; q = 2, 3, \ldots Q - 1. \tag{5.10}$$

We now construct z_i values for training inputs $\mathbf{x}_i^{(q)}$ by considering the following criteria. If $\mathbf{x}_i^{(q)}$ has similar minimum distances $\kappa(\mathbf{x}_i^{(q)}, q-1)$ and $\kappa(\mathbf{x}_i^{(q)}, q+1)$ (and consequently a high value of $w_{\mathbf{x}_i^{(q)}}$), the resulting z_i value should be closer to c_i, so that intuitively, we consider this pattern as *well located* within its class. If $\kappa(\mathbf{x}_i^{(q)}, q-1)$ and $\kappa(\mathbf{x}_i^{(q)}, q+1)$ are very different (and consequently a low value of $w_{\mathbf{x}_i^{(q)}}$ is obtained), the pattern $\mathbf{x}_i^{(q)}$ is closer to one of these classes and so the corresponding z_i value should be closer to the interval of $\mathcal{Z}$ values of the closest adjacent class, $q-1$ or $q+1$. This idea is formalized in the following expression:

3 This does not in any way hamper generality, as our regressors defining g will be smooth nonlinear functions.

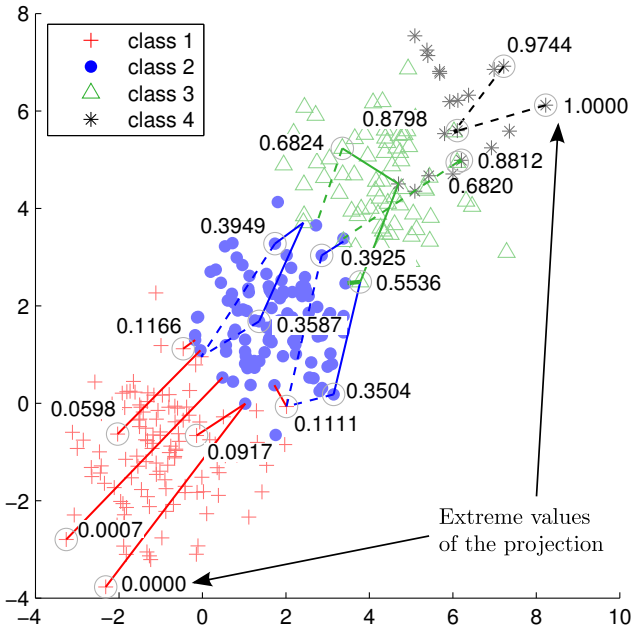


Figure 5.3: Example of the generated z_i values on a synthetic dataset with a linear order relationship.

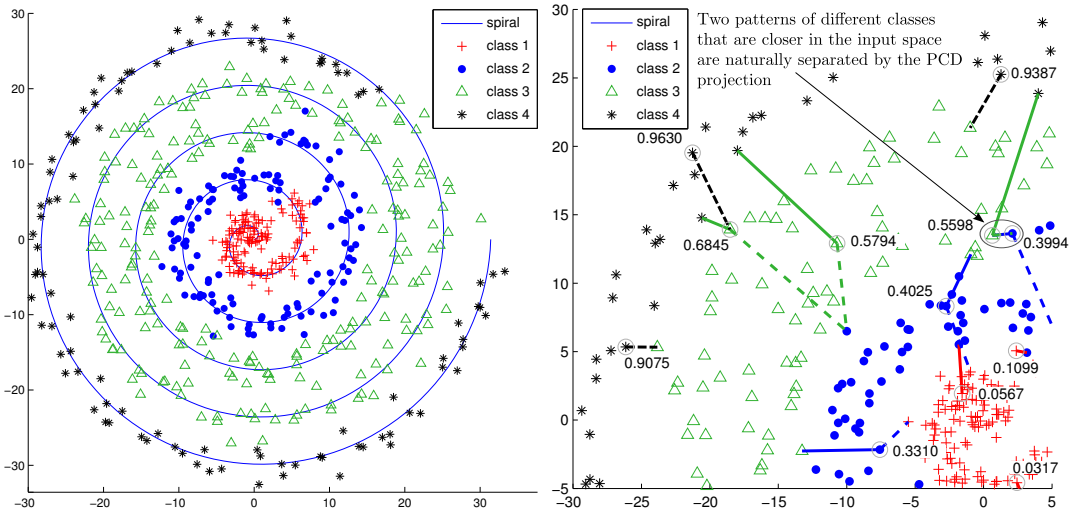
$$z_i = \phi\left(\mathbf{x}_i^{(q)}\right) = \begin{cases} c_1 + (1 - w_{\mathbf{x}_i^{(1)}}) \cdot \frac{1}{Q}, \text{ if } q = 1, \\ c_q - (1 - w_{\mathbf{x}_i^{(q)}}) \cdot \frac{1}{2Q}, \text{ if } q \in \{2, \ldots, Q-1\} \text{ and} \\ \qquad \kappa(\mathbf{x}_i^{(q)}, q-1) \leq \kappa(\mathbf{x}_i^{(q)}, q+1), \\ c_q + (1 - w_{\mathbf{x}_i^{(q)}}) \cdot \frac{1}{2Q}, \text{ if } q \in \{2, \ldots, Q-1\} \text{ and} \\ \qquad \kappa(\mathbf{x}_i^{(q)}, q-1) > \kappa(\mathbf{x}_i^{(q)}, q+1), \\ c_Q - (1 - w_{\mathbf{x}_i^{(Q)}}) \cdot \frac{1}{Q}, \text{ if } q = Q, \end{cases} \tag{5.11}$$

where $w_{\mathbf{x}_i^{(q)}}$ is defined in Eq. (5.8), c_q is the centre of class interval corresponding to C_q (see Eq. (5.10)) and Q is the number of classes. Eq. (5.11) guarantees that all z values lie in the correct class interval[4]. This methodology for data projection is called *Pairwise Class Distances* (PCD).

5.4.4 *Analysis of the proposed projection in synthetic datasets*

For illustration purposes, we generated synthetic ordinal classification datasets in $\mathcal{X} \in \mathbb{R}^2$ with four classes ($Q = 4$). Figure 5.3 shows the patterns of a synthetic dataset, *SyntheticLinearOrder*, with a linear order between classes, whereas Figure 5.4 shows *SyntheticNonLinearOrder* dataset, with a nonlinear ordinal relationship between classes. Points at *SyntheticLinearOrder* were generated by adding an uniform noise to points of a line. Points in *SyntheticNonLinearOrder* were generated by adding a

4 Recall that the threshold set θ delimiting class intervals is defined in Eq. (5.9).



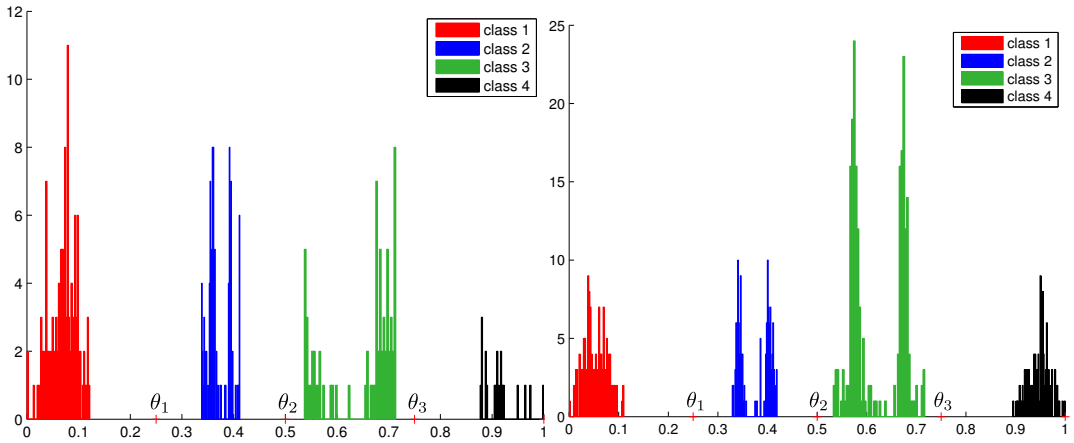
(a) General disposition of the points of the dataset. (b) Upper left area of the dataset.

Figure 5.4: Example of the generated z_i values on the synthetic dataset with a nonlinear class order structure. Figure on the right shows a zooming over the upper left area at the center of the dataset shown on the left.

Gaussian noise to points on a spiral. In both figures, points belonging to different classes are marked with different colours and symbols. Besides the points, the figures also illustrate basic concepts of the proposed method on example points (surrounded by grey circles). For these points, the minimum distances are illustrated with lines of the corresponding class colour. The minimum distance of a point to the previous and next class patterns are marked with dashed and solid lines, respectively. For selected points we show the value of the PCD projection (calculated using Eq. (5.11)).

In Figure 5.3 it can be seen that the z value increases for patterns of the higher classes, and this value varies depending of the position of the pattern $\mathbf{x}^{(q)}$ in the space with respect to the patterns $\mathbf{x}^{(q-1)}$ and $\mathbf{x}^{(q+1)}$ of adjacent classes. Extreme values, $z = 0.0$ and $z = 1.0$ correspond to the patterns more distant from the classes 1 and Q respectively (and with a maximum $w_{\mathbf{x}^{(q)}}$ value). *SyntheticNonLinearOrder* in Figure 5.4 is designed to demonstrate that the PCD projection is suitable for more complex ordinal topologies of the data. This is, for any topology in a ordinal dataset, it is expected that patterns of classes $q-1$ and $q+1$ are always the closest ones to the patterns of class q, and PCD will take advantage from this situation to decide the relative order of the pattern within its class, even when this is produced in a nonlinear manner.

Figure 5.5a and Figure 5.5b show histograms of the PCD projections from the synthetic datasets in Figure 5.3 and Figure 5.4, respectively. The thresholds θ that divide the z values of the different classes are also included. Observe that the z values of the different classes are clearly separated, and that they are compacted within a range which is always smaller than the range initially indicated by the thresholds. This is due to the scaling of the z values in Eq. (5.8), where the $w_{\mathbf{x}^{(q)}}$ value cannot be zero, so a pattern can never be located 'close' to the boundary separating intervals of adjacent classes.



(a) *SyntheticLinearOrder* dataset. (b) *SyntheticNonLinearOrder* dataset.

Figure 5.5: Histograms of the PCD projection of the synthetic datasets.

5.4.5 *Algorithm for ordinal classification*

PCDOC Training: $\{g, \theta\} = \text{PCDOCtr}(\mathbf{D})$.
Require: Training dataset $\mathbf{D}$.
Ensure: Regressor (g) and thresholds (θ).
1: Calculate thresholds θ and centres $\mathbf{c}$ according to Eq. (5.9) and Eq. (5.10).
2: For each pattern, calculate z_i according to Eq. (5.11): $z_i = \phi(\mathbf{x}_i^{(q)})$.
3: Build a regressor g, considering z as the regression response variable: $z = g(\mathbf{x})$.
4: **return** $\{g, \theta\}$

Figure 5.6: PCDOC regression training algorithm pseudocode.

PCDOC Prediction: $\hat{y} = \text{PCDOCpr}(\mathbf{x}, g, \theta)$.
Require: Regressor (g), thresholds (θ) and test input ($\mathbf{x}$).
Ensure: Predicted label ($\hat{y}$).
1: Predict the latent variable value using the regressor g: $\hat{z} = g(\mathbf{x})$.
2: Map the $\hat{z}$ value to the corresponding class using f as defined in Eq. (5.6): $\hat{y} = f(\hat{z}, \theta)$.
3: **return** $\hat{y}$

Figure 5.7: PCDOC classification algorithm for unseen data.

Once the PCD projections have been obtained for all training inputs, we construct a new training set $\mathbf{T}' = \left\{ (\mathbf{x}_i, \phi(\mathbf{x}_i^{(y_i)}) \mid (\mathbf{x}_i, y_i) \in \mathbf{D} \right\}$. Any generic regression tool can be trained on $\mathbf{T}'$ to obtain the projection function $g : \mathcal{X} \to \mathcal{Z}$. In this respect, our method is quite general, allowing the user to choose his or her favorite regression method or any other improved regression tool introduced in the future. The resulting algorithm, named Pairwise Class Distances for Ordinal Classification (PCDOC), is described in two steps in Figure 5.6 and Figure 5.7.

It is expected that formulating the problem as a regression problem would help the model to capture the ordinal structure of the input[5] and output spaces, and their relationship. In addition, due to the nature of the regression problem, it is expected that the performance of the classification task will be improved regarding metrics that consider the difference between the predicted and actual classes within the linear class order, such as MAE or $AMAE$, or the correlation between the target and predicted values, such as τ_b. Experimental results confirm this hypothesis in Section 5.4.7.3.

5.4.6 *Experiments with synthetic data*

This section addresses the PCDOC performance analysis in some controlled experiments, i.e. with controlled synthetically generated data. For this purpose we have developed a specific synthetic data generation tool which is motivated in the following section.

5.4.6.1 *Motivation to the use of synthetic data*

Ideally, dataset selection (real-world or synthetic ones) must test specific issues of a method identified after a complete theoretical analysis. However, this is not the general experimental tendency as Macià has pointed out in Chapter 2 of her PhD. Thesis, where a criticism of experimental assessment of learners based on arbitrary selection of datasets is done [249]. Macià et. al remark the necessity of a proper selection of experimental data guided by data complexity measures [250].

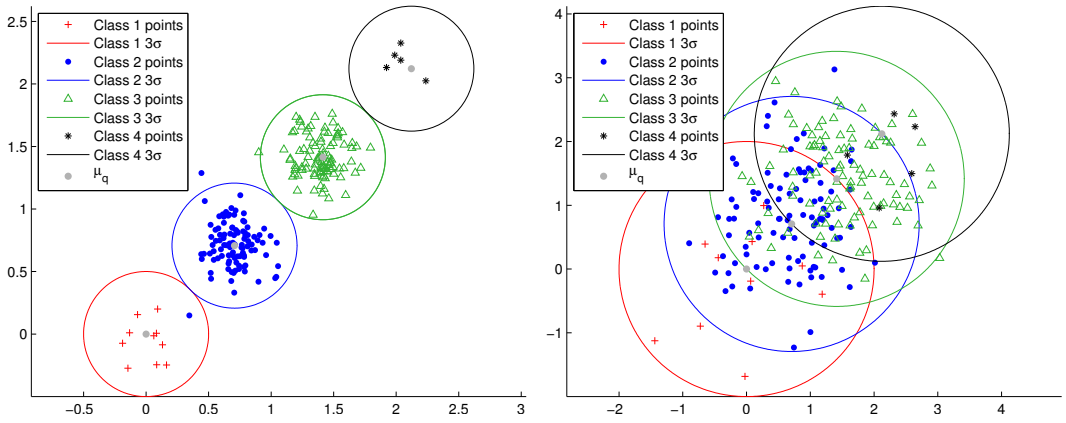
Thereafter, synthetic datasets can be useful in a variety of situations, specifically when new machine learning models and training algorithms are developed or when trying to seek the weaknesses of an specific method. In contrast to real-world data, synthetic datasets provide a controlled environment for analysing concrete critic points such as outlier tolerance, data dimensionality influence and class imbalance, among others.

Even though the relevance of proper synthetic data generation, scientific works on data generation methods based on controlled statistical data properties are scarce [251], as pointed by [252]: "Surprisingly, little work has been done on systematically generating artificial datasets for the analysis and evaluation of data analysis algorithms in data mining area.".

With the above motivation, in this thesis, a framework for synthetic data generation have been developed with special attention to pattern order in the space, data dimensionality, class overlapping and data multimodality. Up to the authors knowledge there is no framework dealing with such a controlled data generator for covering these features. Variables such as position, width and overlapping of data distributions in the n-dimensional space are controlled by considering them as n-spheres. Then, we achieve full control over data topology and over a set of relevant statistical properties of the data. More details about the synthetic data generator are provided at Appendix B and associated publication [242]. The source code of the synthetic data

5 In the case of ordinal regression, the ordinal structure is only a hypothetical about the input data.

generator is available on a public website[6] and it is released under the GNU General Public License version 3 (GPLv3) [253].



(a) Synthetic Gaussian dataset with $\sigma = 0.167$. (b) Synthetic Gaussian dataset with $\sigma = 0.667$.

Figure 5.8: Synthetic Gaussian dataset example for two dimensions.

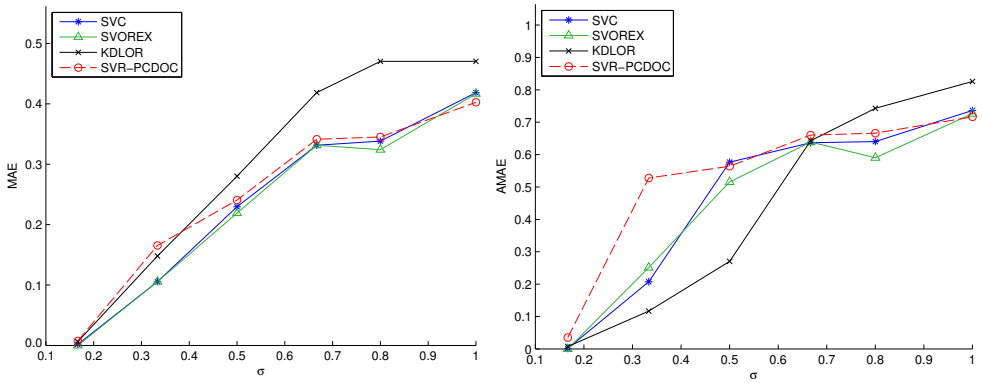
5.4.6.2 *Analysis of the influence of dimensionality and class overlapping.*

This section analyses the performance of the PCDOC algorithm under situations that may hamper classification: class overlapping and large dimensionality of the data. For this purpose, different synthetic datasets have been generated by sampling random points from Q Gaussian distributions, where Q is the number of classes, so that each class points are random samples of the corresponding Gaussian distribution. In order to easily control the overlap of the classes, the variance (σ^2) is kept constant independently of the number of dimensions (K). In addition, the Q centres (means μ_q) are set up in order to keep the distance of 1 between two adjacent class means independently of K. Under this situation, each coordinate of adjacent class means is separated by $\triangle_\mu = 1/\sqrt{K}$ so that $\mu_1 = 0$, $\mu_2 = \mu_1 + \triangle_\mu$, $\mu_3 = \mu_2 + \triangle_\mu$ and so on.

The number of features tested (input space dimensionality) were $K \in \{10, 50, 100\}$ and the different width values are $\sigma \in \{0.167, 0.333, 0.500, 0.667, 0.800, 1.000\}$, so that 18 datasets were generated. The number of patterns for each class from one to four was 10, 100, 100 and 5. Figure 5.8 shows two of these datasets generated with different variance values for $K = 2$.

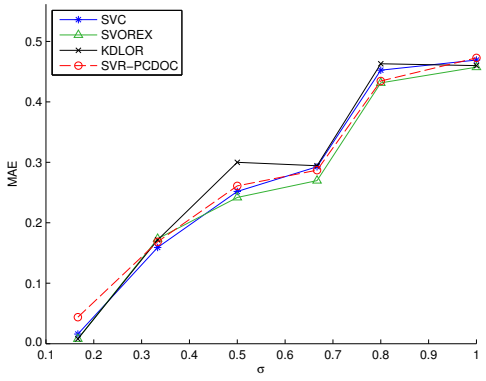
For these experiments, our approach uses the Support Vector Regression (SVR) algorithm as the model for the z variable (the method will be referred to as SVR-PCDOC). We have also included three methods as baseline methods: the C-Support Vector Classification SVC [44, 58], the Support Vector Ordinal Regression with explicit constraints (SVOREX) [33, 39] and the Kernel Discriminant Learning for Ordinal Regression (KDLOR) [89]. As in the next experimental section (Section 5.4.7), the experimental design includes 30 stratified random splits (with 75% of patterns for training and the remaining for generalization). The mean MAE and $AMAE$ generalization results are used for comparison purposes at Figure 5.9. For further details

6 http://www.uco.es/grupos/ayrna/iwann2013-syntheticdatagenerator

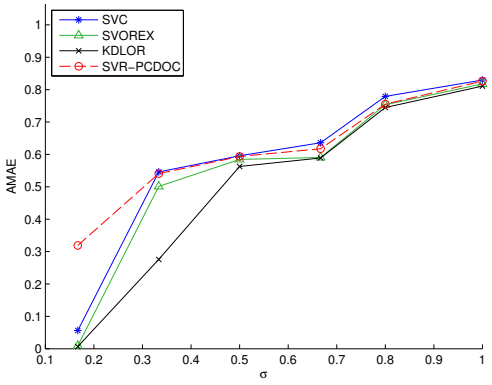


(a) MAE with $K = 10$.

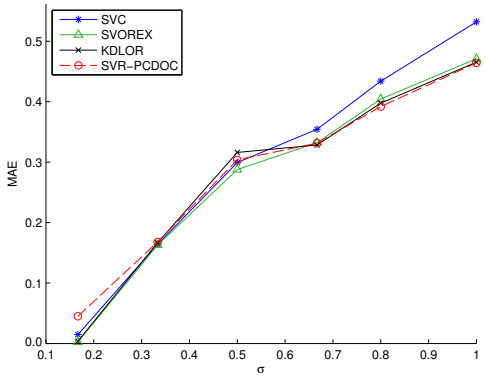
(b) $AMAE$ with $K = 10$.



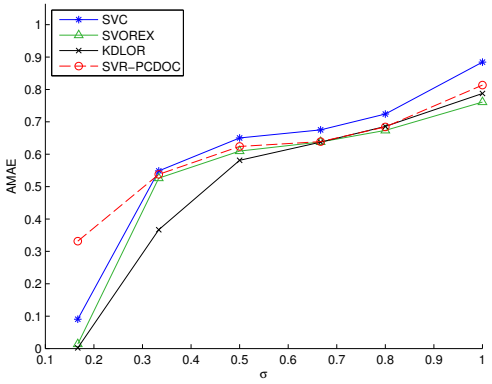
(c) MAE with $K = 50$.



(d) $AMAE$ with $K = 50$.



(e) MAE with $K = 100$.



(f) $AMAE$ with $K = 100$.

Figure 5.9: MAE and $AMAE$ performance for synthetic Gaussian dataset with distribution $\mathbf{x} = \mathcal{N}(\mu, \sigma^2\mathbf{I}_{K\times K})$ and $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^K$, where $\mathbf{I}_{K\times K}$ is the identity matrix.

about experimental procedure and hyper-parameters optimization please refer to the next experimental section (Subsection 5.4.7.2).

From the results depicted at Figure 5.9, we can generally conclude that the three methods, except KDLOR, have similar MAE performance degradation with the increase of class overlapping and dimensionality. Figure 5.9a shows that SVR-PCDOC

have a slightly worse performance than SVC and SVOREX. However, in experiments with higher K (Figures 5.9c and 5.9e) the performance of the three ordinal methods varies in a similar way. In partigular, in Figure 5.9e we can observe that SVC performance decreases with high overlapping and high dimensionality, whereas the ordinal methods have similar performance here. From the analysis of the $AMAE$ performance we can conclude that KDLOR outperforms the rest of the methods in cases of low class overlapping. Regarding our method, we can conclude that compared with the other methods its $AMAE$ performance is worse in the case of low class overlap. However, in general, our method seems more robust when the class overlap increases.

5.4.6.3 *Analysis of the influence of data multimodality.*

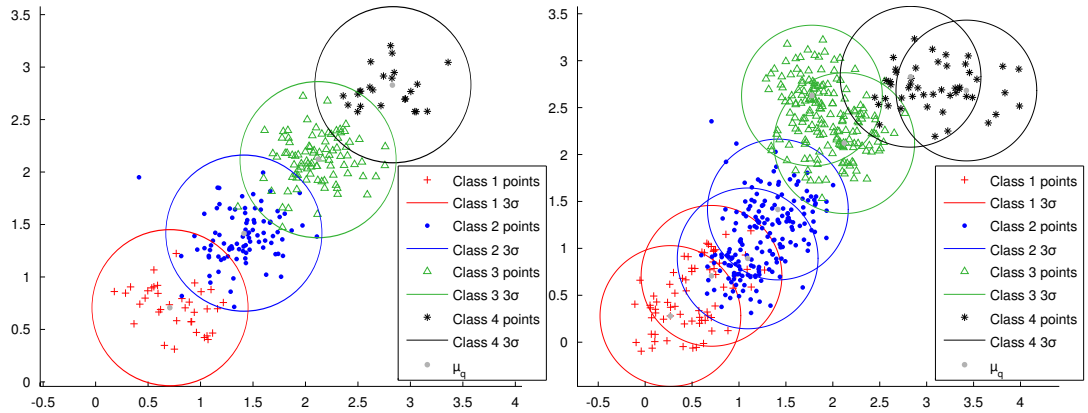
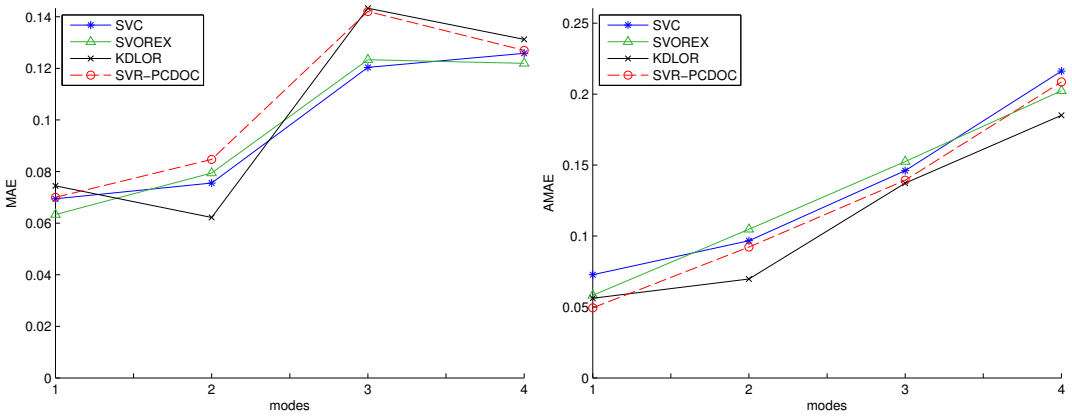


Figure 5.10: Illustration of the unimodal and bimodal cases of the synthetic Gaussian dataset example for $K = 2$ and $\sigma^2 = 0.25$.

This section extends the above experiments to the case of multimodal data, the datasets are generated with $K = 2$ and $\sigma^2 = 0.25$, and the number of modes per class is varied. Figure 5.10a presents the unimodal case. The datasets with more modes per class are generated in the following way. A Gaussian distribution is set up as in the previous section, with center μ_q. For each class, each additional Gaussian distribution is centered in a random location within the hyper-sphere with center μ_q and radius 0.75. Then, patterns are sampled from each distribution. For each class, we considered different number of modes, from one mode to four modes. The number of patterns generated for each mode was 36, 90, 90 and 24 for class 1, 2, 3 and 4, respectively, using the same number for all modes of a class. An example of the bimodal case (two Gaussian distributions per class) is shown in Figure 5.10b, having 72, 180, 180 and 48 patterns for class 1, 2, 3 and 4, respectively.

Experiments were carried out as in the previous section, and MAE and $AMAE$ generalization results are depicted in Figure 5.11. Regarding MAE, Figure 5.11a reveals that the fours methods perform similarly in datasets with one and four modes but they differ on performance for the two and three modes. Only considering MAE, SVRPCDOC has the worse performance in case two and three. Nevertheless, con-



(a) *MAE* performance. (b) *AMAE* performance.

Figure 5.11: *MAE* and *AMAE* performance for synthetic Gaussian dataset with distribution $\mathbf{x} = \mathcal{N}(\mu, \sigma^2\mathbf{I}_{K\times K})$ and $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^K$, where $\mathbf{I}_{K\times K}$ is the identity matrix (being $K = 2$ and $\sigma^2 = 0.25$ for all the synthetic datasets).

sidering *AMAE* results at Figure 5.11b, SVRPCDOC and KDLOR achieve the best results. The different behaviour of the methods depending on the performance measure can be explained by observing the nature of the bimodal dataset (see Figure 5.10b), where the majority of the patterns are from classes two and three. In this context, the optimization done by SVOREX and SVC can move the decision thresshholds to better classify patterns of these two classes at the expense of misclassifying class one and four patterns, especially patterns of those classes placed on the class boundaries (see Figure 5.10b).

5.4.7 *Experiments with real problems*

In this section we report on extensive experiments that were performed to check the competitiveness of the proposed methodology. The source code of PCDOC is available on a public website[7] and it is released under the GNU General Public License version 3 (GPLv3) [253]. The website also includes synthetic datasets analysis code and real ordinal data sets partitions used for the experiments.

5.4.7.1 *Ordinal classification datasets and experimental design*

To the best of our knowledge, there are no public specific datasets repositories for real ordinal classification problems. The ordinal regression benchmark datasets repository provided by Chu et. al [41] is the most widely used repository in the literature. However, these datasets are not real ordinal classification datasets but regression ones. To turn regression into ordinal classification, the target variable was discretized into Q different bins (representing classes), with equal frequency or equal width. However, there are potential problems with this approach. If equal frequency labelling is considered, the datasets do not exhibit some characteristics of typical

7 http://www.uco.es/grupos/ayrna/neco-pairwisedistances

Table 5.9: Datasets used for the experiments (N is the number of patterns, K is the number of attributes and Q is the number of classes).

Dataset	N	K	Q	Ordered Class Distribution
automobile	205	71	6	(3,22,67,54,32,27)
bondrate	57	37	5	(6,33,12,5,1)
contact-lenses	24	6	3	(15,5,4)
eucalyptus	736	91	5	(180,107,130,214,105)
newthyroid	215	5	3	(30,150,35)
pasture	36	25	3	(12,12,12)
squash-stored	52	51	3	(23,21,8)
squash-unstored	52	52	3	(24,24,4)
tae	151	54	3	(49,50,52)
winequality-red	1599	11	6	(10,53,681,638,199,18)

complex classification tasks, such as class imbalance. On the other hand, severe class imbalance can be introduced by using the same binning width. Finally, as the actual target regression variable exists with observed values, the classification problem can be simpler than on those datasets where the variable z is really unobservable and has to be modelled.

We have therefore decided to use a set of real ordinal classification datasets publicly available at the UCI [231] and *mldata.org* repositories [246] (see Table 5.9 for data description). All of them are ordinal classification problems, although one can find literature where the ordering information is discarded. The nature of the target variable is now analysed for two example datasets. *bondrate* dataset is a classification problem where the purpose is to assign the right ordered category to bonds, being the category labels $\{C_1 = \text{AAA}, C_2 = \text{AA}, C_3 = \text{A}, C_4 = \text{BBB}, C_5 = \text{BB}\}$. These labels represent the quality of a bond and are assigned by credit rating agencies, AAA being the highest quality and BB the worst one. In this case, classes AAA, AA, A are more similar than classes BBB and BB so that no assumptions should be done about the distance between classes both in the input and latent space. Other example is *eucalyptus* dataset, in this case the problem is to predict which eucalyptus seedlots are best for soil conservation in a seasonally dry hill country. Being the classes $\{C_1 = \text{none}, C_2 = \text{low}, C_3 = \text{average}, C_4 = \text{good}, C_5 = \text{best}\}$, it cannot be assumed an equal width for each class in the latent space.

Regarding the experimental set up, 30 different random splits of the datasets have been considered, with 75% and 25% of the instances in the training and test sets respectively. The partitions were the same for all compared methods, and, since all of them are deterministic, one model was obtained and evaluated (in the test (generalization) set), for each split. All nominal attributes were transformed into as many binary attributes as the number of categories. All the datasets were property standardized.

5.4.7.2 *Existing methods used for comparison purposes*

For comparison purposes, different state-of-the-art methods have been included in the experimentation. These methods are the following (for methods description see Section 3.4 in Chapter 3):

- Gaussian Processes for Ordinal Regression (GPOR) [41].
- Support vector ordinal regression with implicit (SVORIM) and explicit (SVORIM) constraints [33, 39].
- Reduction from cost-sensitive ordinal ranking to weighted binary classification (RED) with SVM (RED-SVM) [30, 31].
- A Simple Approach to Ordinal Regression (ASAOR) with C4.5 (ASAOR(C4.5)) [27].
- The Proportional Odds Model (POM) [36].
- Kernel Discriminant Learning for Ordinal Regression (KDLOR) [89].
- The Cost Support Vector Classification (SVC) available in libSVM 3.0 [169, 239] is included as a reference nominal classifier.

In our approach, the Support Vector Regression (SVR) algorithm is used as the model for the z variable. The method will be referred to by the acronym SVR-PCDOC. The ϵ-SVR available in libSVM is used.

Model selection is an important issue and involves selecting the best hyper-parameter combination for all the methods compared. All the methods were configured to use the Gaussian kernel. For the support vector algorithms, i.e. SVC, RED-SVM, SVOREX, SVORIM and ϵ-SVR, the corresponding hyper-parameters (regularization parameter, C, and width of the Gaussian functions, γ), were adjusted using a grid search over each of the 30 training sets by a 5-fold nested cross-validation with the following ranges: $C \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$ and $\gamma \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$. Regarding ϵ-SVR, the additional ϵ parameter has to be adjusted. The range considered was $\epsilon \in \{10^{0}, 10^{1}, 10^{2}, 10^{3}\}$. For KDLOR, the width of the Gaussian kernel was adjusted by using the range $\gamma \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$, and the regularization parameter, u, for avoiding the singularity problem values were $u \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$. POM and ASAOR(C4.5) methods have not hyper-parameters. Finally, GPOR-ARD has no hyper-parameters to fix, since the method optimizes the associated parameters itself.

For all the methods, the MAE measure is used as the performance metric for guiding the grid search to be consistent with the authors of the different state-of-the-art methods. The grid search procedure of SVC at libSVM has been modified in order to use MAE as the criteria for hyper-parameters selection.

5.4.7.3 *Performance results*

Tables 5.10 and 5.11 outline the results through the mean and standard deviation (SD) of Acc_{G}, MAE_{G}, $AMAE_{\mathrm{G}}$ and $\tau_{b\mathrm{G}}$ across the 30 hold-out splits, where the subindex G indicates that results were obtained on the (hold-out) generalization fold. As a summary, Table 5.12 shows, for each performance metric, the mean values of the

metrics across all the datasets, and the mean ranking values when comparing the different methods ($R = 1$ for the best performing method and $R = 9$ for the worst one). To enhance readability, in Tables 5.10, 5.11 and 5.12 the best and second-best results are in bold face and italics, respectively.

Table 5.10: Comparison of SVR-PCDOC to other ordinal classification methods and SVC (*Acc* and *MAE*). The mean and standard deviation (SD) of the generalization results are reported for each dataset. The best statistical result is in bold face and the second best result in italics.

	Acc Mean_{SD}									
Method/ DataSet	automobile	bondrate	contact-lenses	eucalyptus	newthyroid	pasture	squash-stored	squash-unstored	tae	winequality-red
ASAOR(C4.5)	$0.696_{0.059}$	$0.533_{0.074}$	$\mathit{0.750}_{\mathit{0.085}}$	$0.639_{0.036}$	$0.917_{0.039}$	$\mathbf{0.752_{0.145}}$	$0.603_{0.118}$	$\mathit{0.774}_{\mathit{0.101}}$	$0.395_{0.058}$	$0.603_{0.021}$
GPOR	$0.611_{0.073}$	$\mathbf{0.578_{0.032}}$	$0.606_{0.093}$	$\mathbf{0.686_{0.034}}$	$0.966_{0.024}$	$0.522_{0.178}$	$0.451_{0.101}$	$0.644_{0.162}$	$0.328_{0.041}$	$0.606_{0.015}$
KLDOR	$\mathbf{0.722_{0.058}}$	$0.542_{0.087}$	$0.589_{0.174}$	$0.611_{0.028}$	$\mathit{0.972}_{\mathit{0.019}}$	$\mathit{0.678}_{\mathit{0.125}}$	$\mathbf{0.703_{0.112}}$	$\mathbf{0.828_{0.104}}$	$0.555_{0.052}$	$0.603_{0.017}$
POM	$0.467_{0.194}$	$0.344_{0.161}$	$0.622_{0.138}$	$0.159_{0.036}$	$\mathit{0.972}_{\mathit{0.022}}$	$0.496_{0.154}$	$0.382_{0.152}$	$0.349_{0.143}$	$0.512_{0.089}$	$0.594_{0.017}$
SVC	$\mathit{0.697}_{\mathit{0.062}}$	$\mathit{0.556}_{\mathit{0.069}}$	$\mathbf{0.794_{0.129}}$	$\mathit{0.653}_{\mathit{0.037}}$	$0.967_{0.025}$	$0.633_{0.134}$	$0.656_{0.127}$	$0.700_{0.082}$	$0.539_{0.062}$	$\mathbf{0.636_{0.021}}$
RED-SVM	$0.684_{0.055}$	$0.553_{0.073}$	$0.700_{0.111}$	$0.651_{0.024}$	$0.969_{0.022}$	$0.648_{0.134}$	$0.664_{0.104}$	$0.749_{0.086}$	$0.522_{0.074}$	$0.618_{0.022}$
SVOREX	$0.665_{0.068}$	$0.553_{0.096}$	$0.650_{0.127}$	$0.647_{0.029}$	$0.967_{0.022}$	$0.630_{0.125}$	$0.628_{0.133}$	$0.718_{0.128}$	$0.581_{0.060}$	$0.629_{0.022}$
SVORIM	$0.639_{0.076}$	$0.547_{0.092}$	$0.633_{0.127}$	$0.639_{0.028}$	$0.969_{0.021}$	$0.667_{0.120}$	$0.639_{0.118}$	$0.764_{0.103}$	$\mathbf{0.590_{0.066}}$	$0.630_{0.022}$
SVR-PCDOC	$0.678_{0.060}$	$0.540_{0.101}$	$0.689_{0.095}$	$0.648_{0.029}$	$\mathbf{0.973_{0.020}}$	$0.656_{0.103}$	$\mathit{0.685}_{\mathit{0.123}}$	$0.695_{0.084}$	$\mathit{0.582}_{\mathit{0.064}}$	$\mathit{0.631}_{\mathit{0.022}}$
	MAE Mean_{SD}									
Method/ DataSet	automobile	bondrate	contact-lenses	eucalyptus	newthyroid	pasture	squash-stored	squash-unstored	tae	winequality-red
ASAOR(C4.5)	$0.401_{0.095}$	$0.624_{0.079}$	$\mathit{0.367}_{\mathit{0.154}}$	$0.384_{0.042}$	$0.083_{0.039}$	$\mathbf{0.248_{0.145}}$	$0.444_{0.140}$	$\mathit{0.239}_{\mathit{0.109}}$	$0.686_{0.146}$	$0.441_{0.023}$
GPOR	$0.594_{0.131}$	$0.624_{0.062}$	$0.511_{0.175}$	$\mathbf{0.331_{0.038}}$	$0.034_{0.024}$	$0.489_{0.190}$	$0.626_{0.148}$	$0.356_{0.162}$	$0.861_{0.155}$	$0.425_{0.017}$
KLDOR	$\mathbf{0.334_{0.076}}$	$0.587_{0.107}$	$0.539_{0.208}$	$0.424_{0.032}$	$\mathit{0.028}_{\mathit{0.019}}$	$\mathit{0.322}_{\mathit{0.125}}$	$\mathbf{0.308_{0.128}}$	$\mathbf{0.172_{0.104}}$	$0.473_{0.069}$	$0.443_{0.019}$
POM	$0.953_{0.687}$	$0.947_{0.321}$	$0.533_{0.241}$	$2.029_{0.070}$	$\mathit{0.028}_{\mathit{0.022}}$	$0.585_{0.204}$	$0.813_{0.248}$	$0.826_{0.230}$	$0.626_{0.126}$	$0.439_{0.019}$
SVC	$0.446_{0.095}$	$0.624_{0.090}$	$\mathbf{0.311_{0.222}}$	$0.394_{0.042}$	$0.033_{0.025}$	$0.367_{0.134}$	$0.377_{0.160}$	$0.308_{0.090}$	$0.578_{0.083}$	$0.408_{0.020}$
RED-SVM	$\mathit{0.393}_{\mathit{0.073}}$	$0.598_{0.088}$	$0.378_{0.169}$	$\mathit{0.380}_{\mathit{0.027}}$	$0.032_{0.022}$	$0.359_{0.142}$	$0.346_{0.110}$	$0.251_{0.086}$	$0.515_{0.087}$	$0.419_{0.021}$
SVOREX	$0.408_{0.089}$	$\mathit{0.573}_{\mathit{0.121}}$	$0.489_{0.185}$	$0.392_{0.031}$	$0.033_{0.022}$	$0.370_{0.125}$	$0.382_{0.139}$	$0.282_{0.128}$	$0.485_{0.078}$	$0.408_{0.023}$
SVORIM	$0.424_{0.090}$	$0.591_{0.102}$	$0.506_{0.167}$	$0.395_{0.035}$	$0.031_{0.021}$	$0.333_{0.120}$	$0.372_{0.126}$	$\mathit{0.239}_{\mathit{0.109}}$	$\mathit{0.461}_{\mathit{0.081}}$	$\mathit{0.406}_{\mathit{0.022}}$
SVR-PCDOC	$0.397_{0.093}$	$\mathbf{0.568_{0.126}}$	$\mathit{0.367}_{\mathit{0.154}}$	$0.392_{0.038}$	$\mathbf{0.027_{0.020}}$	$0.348_{0.104}$	$\mathit{0.326}_{\mathit{0.141}}$	$0.305_{0.084}$	$\mathbf{0.457_{0.071}}$	$\mathbf{0.400_{0.023}}$

Table 5.11: Comparison of SVR-PCDOC to other ordinal classification methods and SVC ($AMAE$ and τ_b). The mean and standard deviation (SD) of the generalization results are reported for each dataset. The best statistical result is in bold face and the second best result in italics.

	$AMAE$ Mean_{SD}									
Method / DataSet	automobile	bondrate	contact-lenses	eucalyptus	newthyroid	pasture	squash-stored	squash-unstored	tae	winequality-red
ASAOR(C4.5)	$0.511_{0.104}$	$1.226_{0.175}$	*$0.315_{0.124}$*	$0.428_{0.045}$	$0.115_{0.056}$	**$0.248_{0.145}$**	$0.502_{0.192}$	**$0.256_{0.149}$**	$0.689_{0.151}$	*$1.045_{0.080}$*
GPOR	$0.792_{0.200}$	$1.360_{0.122}$	$0.651_{0.286}$	**$0.362_{0.040}$**	$0.062_{0.049}$	$0.489_{0.190}$	$0.797_{0.234}$	$0.443_{0.226}$	$0.863_{0.164}$	$1.065_{0.065}$
KLDOR	**$0.345_{0.104}$**	*$1.037_{0.270}$*	$0.519_{0.280}$	$0.426_{0.038}$	$0.059_{0.040}$	*$0.322_{0.125}$*	**$0.349_{0.156}$**	*$0.309_{0.180}$*	$0.471_{0.070}$	$1.258_{0.069}$
POM	$1.026_{0.800}$	$1.103_{0.403}$	$0.535_{0.275}$	$1.990_{0.048}$	*$0.050_{0.040}$*	$0.585_{0.204}$	$0.815_{0.251}$	$0.791_{0.332}$	$0.627_{0.128}$	$1.085_{0.037}$
SVC	$0.486_{0.125}$	$1.265_{0.183}$	**$0.307_{0.277}$**	$0.433_{0.048}$	$0.060_{0.051}$	$0.367_{0.134}$	$0.446_{0.189}$	$0.444_{0.163}$	$0.576_{0.083}$	$1.119_{0.069}$
RED-SVM	$0.468_{0.096}$	$1.184_{0.225}$	$0.385_{0.198}$	$0.414_{0.030}$	$0.057_{0.049}$	$0.359_{0.142}$	$0.391_{0.149}$	$0.348_{0.159}$	$0.513_{0.086}$	$1.068_{0.069}$
SVOREX	$0.518_{0.096}$	$1.072_{0.217}$	$0.517_{0.303}$	$0.411_{0.034}$	$0.054_{0.042}$	$0.370_{0.125}$	$0.433_{0.172}$	$0.426_{0.157}$	$0.484_{0.079}$	$1.095_{0.067}$
SVORIM	$0.523_{0.105}$	$1.114_{0.233}$	$0.589_{0.259}$	$0.420_{0.043}$	$0.055_{0.042}$	$0.333_{0.120}$	$0.427_{0.148}$	$0.367_{0.140}$	*$0.459_{0.081}$*	$1.093_{0.072}$
SVR-PCDOC	*$0.440_{0.128}$*	**$0.969_{0.224}$**	$0.420_{0.098}$	*$0.400_{0.043}$*	**$0.045_{0.040}$**	$0.348_{0.104}$	*$0.360_{0.184}$*	$0.396_{0.158}$	**$0.455_{0.071}$**	**$1.040_{0.096}$**
	τ_b Mean_{SD}									
Method / DataSet	automobile	bondrate	contact-lenses	eucalyptus	newthyroid	pasture	squash-stored	squash-unstored	tae	winequality-red
ASAOR(C4.5)	$0.741_{0.069}$	$0.143_{0.159}$	*$0.604_{0.216}$*	*$0.802_{0.025}$*	$0.853_{0.067}$	**$0.778_{0.132}$**	$0.415_{0.245}$	*$0.692_{0.145}$*	$0.243_{0.177}$	$0.496_{0.036}$
GPOR	$0.557_{0.118}$	$0.000_{0.000}$	$0.348_{0.304}$	**$0.830_{0.020}$**	$0.938_{0.045}$	$0.461_{0.314}$	$0.075_{0.211}$	$0.420_{0.331}$	$-0.018_{0.108}$	$0.523_{0.026}$
KLDOR	**$0.793_{0.056}$**	$0.356_{0.257}$	$0.448_{0.273}$	$0.786_{0.017}$	$0.948_{0.034}$	*$0.718_{0.133}$*	**$0.646_{0.160}$**	**$0.764_{0.161}$**	$0.477_{0.114}$	$0.460_{0.028}$
POM	$0.495_{0.283}$	$0.290_{0.302}$	$0.458_{0.309}$	$0.008_{0.038}$	*$0.949_{0.040}$*	$0.463_{0.237}$	$0.169_{0.304}$	$0.109_{0.305}$	$0.317_{0.129}$	$0.497_{0.025}$
SVC	$0.695_{0.077}$	$0.121_{0.177}$	$0.601_{0.300}$	$0.783_{0.025}$	$0.939_{0.045}$	$0.698_{0.133}$	$0.541_{0.240}$	$0.599_{0.140}$	$0.375_{0.110}$	$0.516_{0.027}$
RED-SVM	*$0.751_{0.054}$*	$0.254_{0.247}$	$0.577_{0.242}$	$0.800_{0.017}$	$0.943_{0.041}$	$0.707_{0.129}$	$0.601_{0.148}$	$0.662_{0.108}$	$0.417_{0.120}$	$0.525_{0.030}$
SVOREX	$0.749_{0.062}$	*$0.369_{0.216}$*	$0.425_{0.304}$	$0.794_{0.019}$	$0.941_{0.040}$	$0.691_{0.115}$	$0.534_{0.207}$	$0.592_{0.212}$	$0.445_{0.110}$	$0.531_{0.028}$
SVORIM	$0.748_{0.065}$	$0.299_{0.230}$	$0.382_{0.269}$	$0.792_{0.020}$	$0.944_{0.038}$	$0.710_{0.114}$	$0.542_{0.167}$	$0.656_{0.187}$	*$0.482_{0.118}$*	*$0.533_{0.030}$*
SVR-PCDOC	$0.744_{0.076}$	**$0.455_{0.218}$**	**$0.620_{0.217}$**	$0.795_{0.024}$	**$0.952_{0.037}$**	$0.712_{0.102}$	*$0.610_{0.201}$*	$0.602_{0.133}$	**$0.493_{0.101}$**	**$0.542_{0.033}$**

Table 5.12: Mean results of accuracy ($\overline{Acc}_G$), MAE ($\overline{MAE}_G$), $AMAE$ ($\overline{AMAE}_G$) and τ_b ($\overline{\tau_b}_G$) and mean ranking ($\overline{R}_{Acc_G}$, $\overline{R}_{MAE_G}$, $\overline{R}_{AMAE_G}$ and $\overline{R}_{\tau_b G}$) for the generalization sets.

Method/Metric	$\overline{Acc}_G$	$\overline{R}_{Acc_G}$	$\overline{MAE}_G$	$\overline{R}_{MAE_G}$	$\overline{AMAE}_G$	$\overline{R}_{AMAE_G}$	$\overline{\tau_b}_G$	$\overline{R}_{\tau_b G}$
GPOR	0.666	5.40	0.392	5.25	0.534	4.90	0.577	5.20
ASAOR(C4.5)	0.600	6.50	0.485	7.00	0.688	7.00	0.413	7.50
KDLOR	*0.680*	4.20	*0.363*	*4.00*	*0.509*	*3.80*	*0.640*	*3.60*
POM	0.490	7.90	0.778	7.80	0.861	7.00	0.375	6.90
SVC	**0.683**	**3.60**	0.385	5.55	0.550	6.20	0.587	6.20
RED-SVM	0.676	4.15	0.367	*4.00*	0.519	4.10	0.624	4.00
SVOREX	0.667	5.05	0.382	4.75	0.538	4.90	0.607	5.00
SVORIM	0.672	4.40	0.376	4.05	0.538	4.80	0.609	4.20
SVR-PCDOC	0.678	*3.80*	**0.359**	**2.60**	**0.487**	**2.30**	**0.653**	**2.40**

Regarding Tables 5.10 and 5.11, it can be seen that the majority of methods are very competitive. The best performing method depends on the considered performance metric, as it can be seen from the mean rankings. This is also true when separately considering each of the datasets, and the performance for some datasets varies noticeably if $AMAE_G$ is considered instead of MAE_G (see *bondrate*, *contact-lenses*, *eucalyptus*, *squash-unstored* and *winequality-red*). In the case of *winequality-red*, it happens that the second worse method in MAE_G, ASAOR(C4.5), is the second best one for $AMAE_G$. It is worthwhile to mention that for *pasture* dataset the mean MAE_G and $AMAE_G$ are the same, which is due to the fact that *pasture* is a perfectly balanced dataset (see metrics definition in Section 3.2.4). In the case of *tae*, MAE_G and $AMAE_G$ are very similar since the patterns distribution across classes is very similar. Regarding τ_{bG}, it is interesting to highlight that a value close to zero of this metric reveals that the classifier predictions are not related to the real values, this is, the classifier performance is similar to the performance of a trivial classifier. This happens for GPOR method in the *bondrate*, *squash-stored* and *tae* datasets, and for POM in the *eucalyptus* dataset.

From Table 5.12, it can be observed how best mean value across the different datasets is not always translated into best mean ranking (see $\overline{Acc}_G$ and $\overline{R}_{Acc_G}$ columns). We now analyze the results in greater detail, highlighting the best and second best performances. When considering Acc_G, SVC is clearly the best method, both in average performance and ranking. KDLOR and SVR-PCDOC are the second best methods, in average value and ranking, respectively. However, results are very different for all the other measures, where the order is included in the evaluation. The best method in average MAE_G and ranking of MAE_G is SVR-PCDOC and the second best ranks are for KDLOR and RED-SVM, having similar mean MAE_G. $AMAE$ is a better alternative than MAE when the distribution of patterns is not balanced, and this is clearly the case for several datasets (see Table 5.9). The best values for mean $AMAE_G$ and mean ranking are obtained by SVR-PCDOC, and the second ones are those reported by KDLOR. Finally, the τ_{bG} is revealing the most clear differences. When using this metric, the best mean values and ranks are reported by SVR-PCDOC, followed by KDLOR.

Table 5.13: Differences and Critical Difference (CD) value in rankings in the Bonferroni-Dunn test, using SVR-PCDOC as the control method.

	Method							
Metric	ASAOR(C4.5)	GPOR	KDLOR	POM	SVC	RED-SVM	SVOREX	SVORIM
Acc	1.60	2.70	0.40	4.10•	0.20	0.35	1.25	0.60
MAE	2.65	4.40•	1.40	5.20•	2.95	1.40	2.15	1.45
$AMAE$	2.60	4.70•	1.50	4.70•	3.90•	1.80	2.60	2.50
τ_b	2.80	5.10•	1.20	4.50•	3.80•	1.60	2.60	1.80

Bonferroni-Dunn Test: $CD_{(\alpha=0.05)} = 3.336$

•: Statistical difference with $\alpha = 0.05$

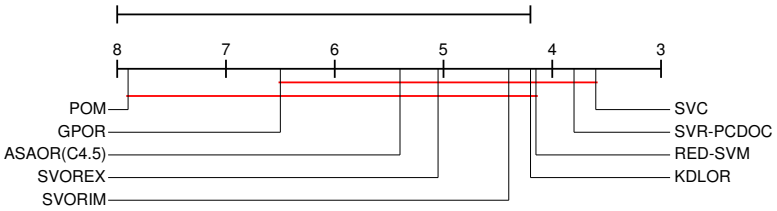
5.4.7.4 *Statistical comparisons between methods*

To quantify whether a statistical difference exists between any of these algorithms, a procedure for comparing multiple classifiers over multiple datasets is employed [240]. First of all, a Friedman's non-parametric test [247] with a significance level of $\alpha = 0.05$ has been carried out to determine the statistical significance of the differences in the mean ranks of Table 5.12 for each different measure. The test rejected the null-hypothesis stating that the differences in mean rankings of Acc_G, MAE_G, $AMAE_G$ and τ_{bG} obtained by the different algorithms were statistically significant (with $\alpha = 0.05$). Specifically, the confidence interval for this number of datasets and algorithms is $C_0 = (0, F_{(\alpha=0.05)} = 2.070)$, and the corresponding F-value for each metric were $3.257 \notin C_0$, $4.821 \notin C_0$, $4.184 \notin C_0$ and $5.099 \notin C_0$, respectively.

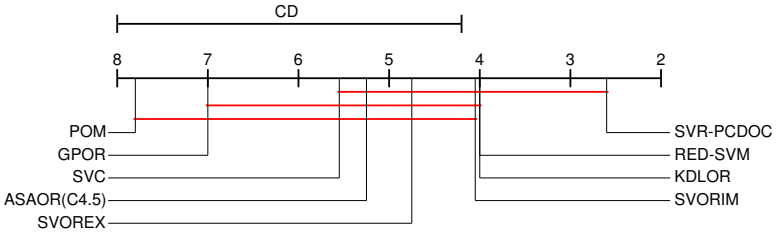
On the basis of this rejection, the Nemenyi post-hoc test is used to compare all classifiers to each other [240]. This test considers that the performance of any two classifiers is deemed significantly different if their mean ranks differ by at least the critical difference (CD), which depends on the number of datasets and methods. A 5% significance confidence was considered ($\alpha = 0.05$) to obtain this CD and the results can be observed in Figure 5.12, which shows CD diagrams as proposed in [240]. Each method is represented as a point in a raking scale, corresponding to its mean ranking performance. CD segments are included to measure the separation needed between methods in order to assess statistical differences. Red lines group algorithms were statistically different mean ranking performance cannot be assessed. Table 5.12 should also be considered when interpreting this graph.

Figure 5.12a shows that SVC, i.e. the nominal classifier, has the best performance in Acc, this is, when not considering the order of the label prediction errors, and SVR-PCDOC has the second best one. RED-SVM, KDLOR and SVORIM have similar performance here. In Figure 5.12b, the best mean ranking is for SVR-PCDOC, and SVORIM, KDLOR and RED-SVM have similar performance. However, when considering $AMAE$, it can be seen at Figure 5.12c that SVR-PCDOC mean ranking distance to the other methods increases, specifically for RED-SVM and SVORIM. Finally, Figure 5.12d shows the mean rank CD diagram for τ_b where SVR-PCDOC is still having the best mean performance.

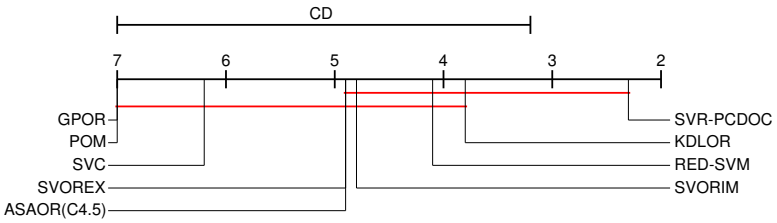
It has been noticed that the Nemenyi approach comparing all classifiers to each other in a post-hoc test is not as sensitive as the approach comparing all classifiers



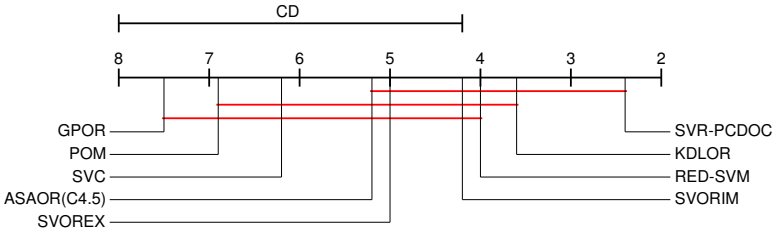
(a) Nemenyi CD diagram comparing generalization Acc mean rankings of the different methods.



(b) Nemenyi CD diagram comparing the generalization MAE mean rankings of the different methods.



(c) Nemenyi CD diagram comparing the generalization $AMAE$ mean rankings of the different methods.



(d) Nemenyi CD diagram comparing the generalization τ_b mean rankings of the different methods.

Figure 5.12: Ranking test diagrams for the mean generalization Acc, MAE, $AMAE$ and τ_b.

to a given classifier (a control method) [240]. The Bonferroni-Dunn test allows this latter type of comparison and, in our case, it is done using the proposed method as the control method for the four metrics. The results of the Bonferroni-Dunn test for $\alpha = 0.05$ can be seen in Table 5.13, where the corresponding critical values are included. From the results of this test, it can be concluded than SVR-PCDOC does not report a statistically significant difference with respect to the SVM ordinal regression methods, KDLOR and ASAOR(C4.5), but it does when it is compared to POM for all the metrics and compared to GPOR for the ordinal metrics. Moreover, there are significant differences with respect to SVC, when considering $AMAE$ and τ_b.

From the above experiments, we can conclude that the reference (baseline) nominal classifier, SVC, is improved with statistical differences when considering ordinal classification measures. Regarding ASAOR(C4.5), SVOREX, SVORIM, KDLOR and RED-SVM, whereas the general performance is slightly better, there are no statistically significant differences favouring any of the methods.

As a summary of the experiments, two important conclusions can be drawn about the performance measures: When imbalanced datasets are considered, Acc_G is clearly omitting important aspects of ordinal classification and so does MAE_G. If the comparative performance is taken into account, KDLOR and SVR-PCDOC appear to be very good classifiers when the objective is to improve $AMAE_G$ and τ_G. The best mean ranking performance is obtained by PCDOC.

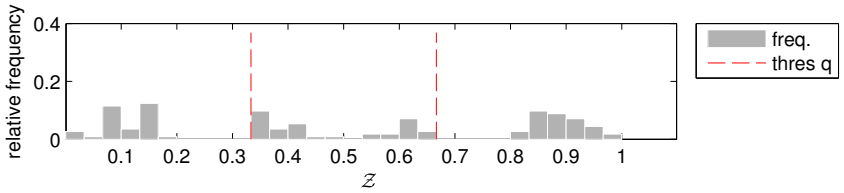
5.4.7.5 *Latent space representations of the ordinal classes*

In the previous section we have shown that our simple and intuitive methodology can compete on equal footing with established more complex and/or less direct methods for ordinal classification. In this section we complement this performance based comparison with a deeper analysis of the main ingredient of our and other related approaches to ordinal classification - projection onto the one-dimesional (latent) space naturally representing the ordinal nature of the class organization. In particular, we study how nonlinear latent variable models, SVR-PCDOC, KDLOR, SVOREX and SVORIM organize their one-dimensional latent space data projections. For comparison purposes, the latent variable $\mathcal{Z}$ values of the training and generalization data of the first fold of *tae* dataset are shown (Figure 5.13). Both histograms and values are plotted so that the behaviour of the models can be analysed. In the case of PCDOC, the PCD projection is also included to see whether the regressor model is close to the PCDOC projection. The histograms represent relative frequency of the projections. SVORIM histograms and latent variable values are not presented since they are similar to the SVOREX ones in the selected dataset.

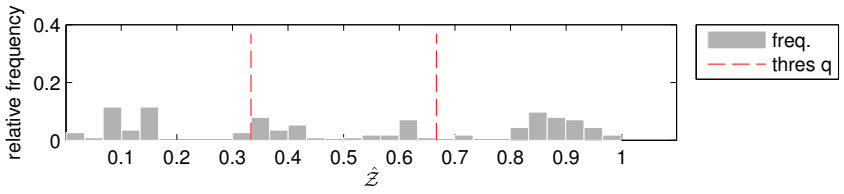
We first analyse the SVR-PCDOC method. From PCD projections in Figure 5.13a we deduce that classes C_1 and C_2 contain patterns that are very close in the input space – projection of some patterns from C_2 lies near the threshold that divides the $\mathcal{Z}$ values for the two classes. Analogous comment applies to classes C_2 and C_3. The regressor seems to have learnt the imposed projection reasonably well since the predicted latent values have a histogram similar to the training PCD projection histogram. The generalization PCD projections (Figure 5.13c) have similar characteristics as the training ones[8]. Note the concentration of values around $\hat{z} = 0.5$ on the prediction of the generalization $\mathcal{Z}$. This concentration of values is due to wrong prediction of class C_1 and C_3 patterns that were both assigned to C_2. This behaviour can be better seen in Figures 5.13e and 5.13f, where the modelled latent value for each pattern is shown together with its class label. Indeed, during training some C_1 and C_2 patterns were mapped to positions near the thresholds. This is probably caused by noise or overlapping class distribution in the input space.

Figure 5.14 presents latent variable values of KDLOR. The KDLOR method projects the data onto the latent space by minimizing the intra-class distance while maximiz-

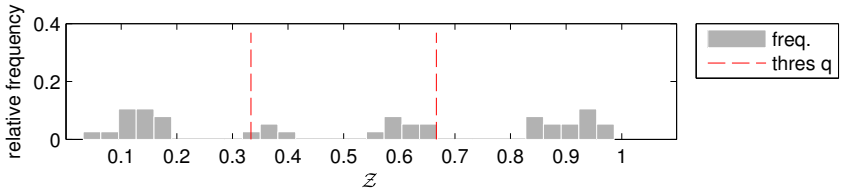
8 There are much less patterns in the hold-out set than in the training set, making direct comparison of the two histograms problematic



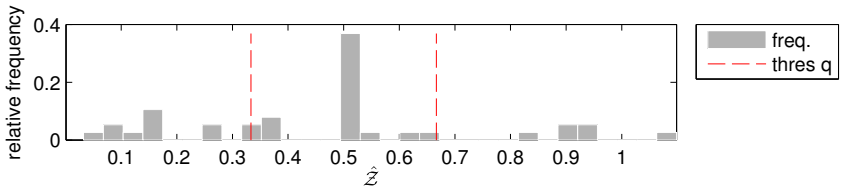
(a) SVR-PCDOC train PCD histogram.



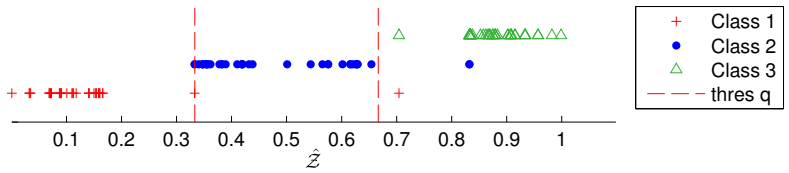
(b) SVR-PCDOC train $\hat{\mathcal{Z}}$ histogram.



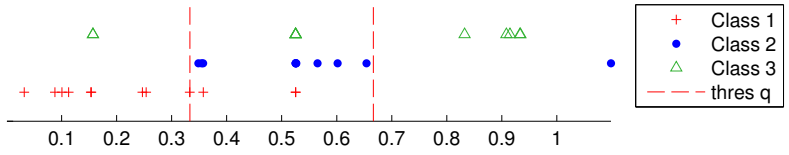
(c) SVR-PCDOC generalization PCD histogram.



(d) SVR-PCDOC generalization $\hat{\mathcal{Z}}$ histogram.



(e) SVR-PCDOC train $\hat{\mathcal{Z}}$ prediction.

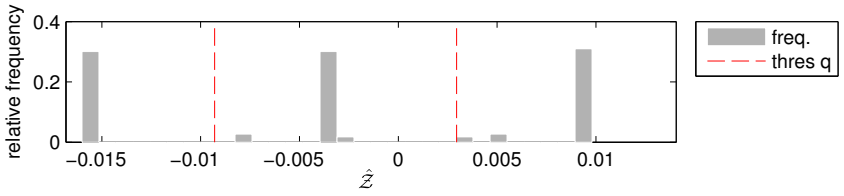


(f) SVR-PCDOC generalization $\hat{\mathcal{Z}}$ prediction.

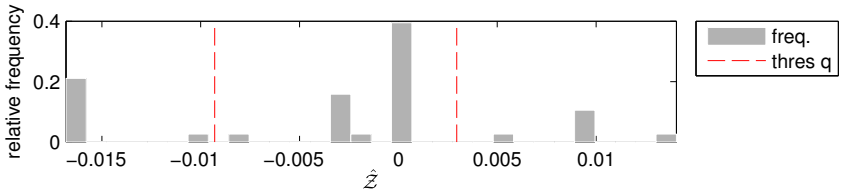
Figure 5.13: PCD projection and SVR-PCDOC's histograms and $\hat{\mathcal{Z}}$ prediction corresponding to the latent variable of the *tae* dataset: train PCD, train predicted $\hat{\mathcal{Z}}$ by SVR-PCDOC, generalization PCD and generalization predicted $\hat{\mathcal{Z}}$ by SVR-PCDOC. Generalization results are $Acc = 0.582$, $MAE = 0.457$, $AMAE = 0.455$, $\tau_b = 0.493$.

ing the inter-class distance of the projections. As a result, the latent representations of

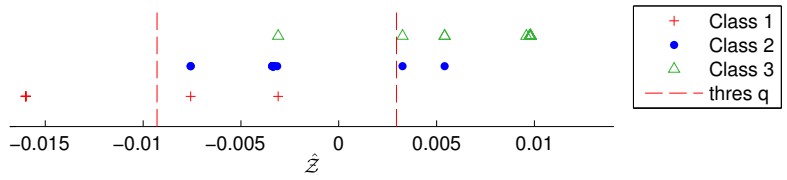
the data are quite compact for each class (see training projection histogram in Figure 5.14a). While this philosophy often leads to superior classification results, the projections are not reflecting the structure of patterns within a single class, that is, the ordinal nature of the data is not fully captured by the model. In addition, KDLOR projections occur in the wrong bins more often than in the case of SVR-PCDOC (see generalization projections $\mathcal{Z}$ in Figure 5.14d)).



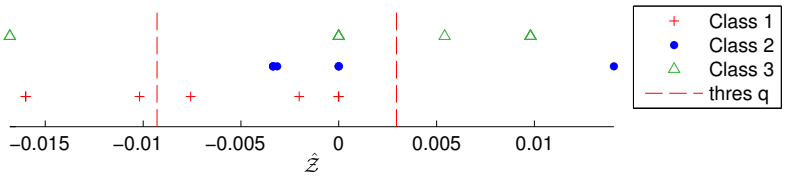
(a) KDLOR train $\hat{\mathcal{Z}}$ histogram.



(b) KDLOR generalization $\hat{\mathcal{Z}}$ histogram.



(c) KDLOR train $\hat{\mathcal{Z}}$ prediction.



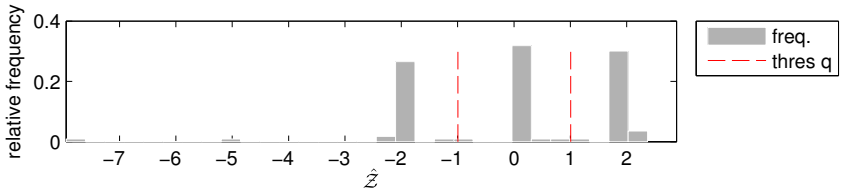
(d) KDLOR generalization $\hat{\mathcal{Z}}$ prediction.

Figure 5.14: Prediction of train and generalization $\hat{\mathcal{Z}}$ values corresponding to KDLOR at *tae* dataset. Generalization results are $Acc = 0.555$, $MAE = 0.473$, $AMAE = 0.471$, $\tau_b = 0.477$.

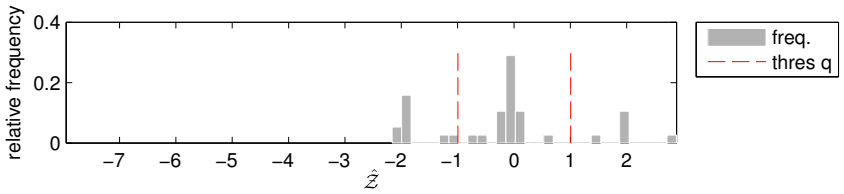
Finally, Figure 5.15 presents latent representations of patterns by the SVOREX model. As in the KDLOR case, (except for a few patterns) the training latent representations are highly compact within each class. Again, the relative structure of patterns within their classes is lost in the projections.

In both models, KDLOR and SVOREX, there is a pressure in the model construction phase to find one-dimensional projections of the data that result in compact classes, while maximizing the inter-class separation. In the case of KDLOR this is explicitly formulated in the objective function. On the other hand, the key idea behind SVM based approaches is margin maximization. Data projections that maximize inter-class margins implicitly make the projected classes compact. We hypothesise that the

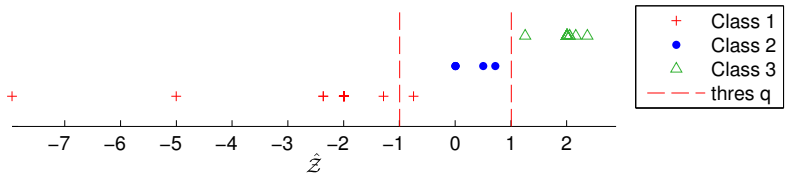
pressure for compact within-class latent projections can lead to poorer generalization performance, as illustrated in Figure 5.15d. In the case of overlapping classes the drive for compact class projections can result in locally highly nonlinear projections of the overlapping regions, over which we do not have a direct control (unlike in the case of PCDOC, where the nonlinear projection is guided by the relative positions of points with respect to the other classes). Having such highly expanding projections can result in test points being projected to wrong classes in an arbitrary manner. Even though we provide detailed analysis for one dataset and one fold only, the observed tendencies were quite general across the data sets and hold-out folds.



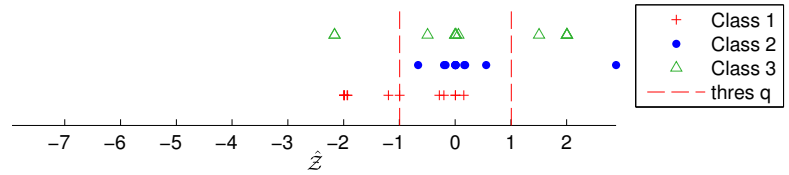
(a) SVOREX train $\hat{z}$ histogram.



(b) SVOREX generalization $\hat{z}$ histogram.



(c) SVOREX train $\hat{z}$ prediction.



(d) SVOREX generalization $\hat{z}$ prediction.

Figure 5.15: Prediction of train and generalization $\hat{z}$ values corresponding to SVOREX at *tae* dataset. Generalization results are $Acc = 0.581$, $MAE = 0.485$, $AMAE = 0.484$, $\tau_b = 0.445$

5.4.8 *Conclusions*

This chapter addresses ordinal classification by proposing a projection of the input data into a one-dimensional variable, based on the relative position of each pattern with respect to the patterns of the adjacent classes. Our approach is based on a simple

and intuitive idea: instead of implicitly inducing a one dimensional data projection into a series of class intervals (as done in threshold based methods), construct such projection explicitly and in a controlled manner. Threshold methods crucially depend on such projections and we propose that it might be advantageous to have a direct control over how the projection is done, rather than having to rely on its indirect induction through a one-stage ordinal classification learning process.

Applying this one-dimensional projection on the training set yields data on which generalized projection can be trained using any standard regression method. The generalized projection can in turn be applied to new instances which are then classified based to the interval into which their projection falls.

We construct the projection by imposing that the *'best separated'* pattern of each class (i.e. the pattern most distant from the adjacent classes) should be mapped to the centre of the interval representing that class (or in the interval extremes for the extreme, first and the last, classes). All the other patterns are proportionally positioned in their corresponding class intervals around the centres mentioned above. We designed a projection method having such desirable properties and empirically verified its appropriateness on datasets with linear and nonlinear class ordering topologies.

We extensively evaluated our method on ten real-world datasets, four performance metrics, a measure of statistical significance and performed comparison with eight alternative methods, including the most recent proposals for ordinal regression and a baseline nominal classifier. In spite of the intrinsic simplicity and straightforward intuition behind our proposal, the results are competitive and consistent with respect to the state-of-the-art in the literature. The mean ranking performance of our method was particularly impressive, when robust ordinal performance metrics were considered, such as the average mean absolute error or the τ_b correlation coefficient. Moreover, we studied in detail the latent space organization of the projection based methods considered in this chapter. We suggest, that while the pressure for compact within class latent projections can make training sample projections nicely compact within classes, it can lead to poorer generalization performance overall.

We also identify some interesting discussion points. Firstly, the latent space thresholds are fixed by the projection with an equal width. This may be interpreted as an assumption of equal widths for each class, which is not always true for all the problems. This would indeed be a problem if we used a linear regressor from the data space to the projection space. However, we employ nonlinear projections and the adjustment for unequal 'widths' of the different classes can be naturally achieved within such nonlinear mapping from the data to the projection space. Actually, from the model fitting standpoint, having fixed-width class regions in the projection space is desirable. Allowing for variable widths would increase the number of free parameters and would make the free parameters dependent in potentially complicated manner (flexibility of projections versus class widths in the projection space). This may have harmful effect on model fitting, especially if the data set is of limited size. Having less free parameters is also advantageous from the point of view of computational complexity.

The second discussion point is the possible undesirable influence of outliers in the PCD projection. One possible solution can be to place each pattern in the projection considering more classes than just the adjacent ones. However, this idea should be done carefully in order not to decrease the role of ordinal information in the

projection. A direct alternative can be to use k-NN like scheme in Eq. (5.7), where instead of taking the minimum distance to a point, the average distance to the k closest points of class $q \pm 1$ can be used. This will represent a generalization of the current scheme that calculates distances with $k = 1$. Nevertheless, the inclusion of k would imply the addition of a new free parameter to the training process.

In conclusion, the results indicate that our two-phase approach to ordinal classification is a viable and simple-to-understand alternative to the state-of-art. The projection constructed in the first phase is consistently extracting useful information for ordinal classification. This last remark is part of the application of the PCDOC to a real world problem in the following chapter.

Part III

APPLICATIONS

This part of the thesis groups two real world problems in which ordinal regression has been applied. The first application is the problem of sovereign debt rating, which our PCDOC classifier is applied to. The second application corresponds to wind speed prediction, which is a fundamental issue for wind farms set up.

6

APPLICATION OF ORDINAL REGRESSION TO SOVEREIGN CREDIT RATING

Summary. This chapter presents an application of the Pairwise Class Distances for Ordinal Classification (PCDOC) technique proposed at Chapter 5.

Specifically, in this chapter, we apply this technique to sovereign credit rating classification, which has had an increasing importance since the beginning of the financial crisis. Credit rating agencies opacity has been criticised by several authors, highlighting the suitability of designing more objective alternative methods. Here we address the sovereign credit rating classification problem within an ordinal classification perspective, including the PCDOC and several ordinal and nominal classification methods.

In addition to the classification task, we use the projection of the regressor model in PCDOC for ranking visualization, which might be suitable to build a decision support system. In contrast to unsupervised visualization and projection techniques, here we have a projection which is validated by means of its suitability to correctly classify patterns.

In this chapter the PCDOC technique is applied to Standard & Poors, Moody's and Fitch sovereign credit rating data of U27 countries during the period 2006-2010.

Submitted publication. Part of this chapter has been submitted to an international journal of application of computational intelligence techniques and is currently under review process:

- J. Sánchez-Monedero, P. Campoy-Muñoz, P.A. Gutiérrez and C. Hervás-Martínez. A guided data projection technique for classification of sovereign ratings: the case of European Union 27. *Applied Soft Computing* (Under Review).

6.1 INTRODUCTION

The sovereign credit rating (or simply sovereign rating) industry is relatively new and has rapidly grown since Standard & Poors (S&P) published the first ranking of sovereign issuers in January 1961, followed by Moody's in 1974 and Fitch in 1994. Rating the creditworthiness of sovereign issuers has drawn growing attention due to the fact that the national governments are by far the largest borrowers in capital markets, outnumbering 60% of debt issued [254]. Sovereign ratings are a condensed assessment of each government's ability and willingness to service its debts in full

and on time [255], distilling a multitude of credit risk information into a single letter on a credit quality scale. The main advantage of the sovereign ratings is the providing of a way of comparing investment and their credit quality to international private investors due to "the lack of consistent standards on government accounting across borders" [254]. Furthermore, in the framework of Basel Accords [256], they play a public function in determining the capital requirements for banks, securities firms and insurance companies according their assets and liabilities [256]. In this way, the role of sovereign ratings in structured finances has been accentuated by both market and regulatory practices.

The European debt crisis is a dramatic example of sovereign rating's role in the today financial market functioning and their economic consequences. The rating downgrade was focused on the so-called PIGS countries, i.e., Portugal, Ireland, Greece and Spain, but led significant spillovers across other European countries with solid macroeconomic and fiscal fundamentals [257]. As a result, the Eurozone financial markets have been under the pressure of the widening of sovereign bond and credit default swap spread, threatening the very existence of the European Union [258].

In the face of these developments, many policymakers and commentators have stated that Credit Rating Agencies (CRAs) precipitated the European crisis by the timing and extent of the downgrades [254]. Their critics highlight some of the disadvantages of the credit risk assessment process carried out by CRAs, such as their inherent conflict of interest within their business model or their adequacy of performance and lack of transparency [256].

The "issuer-pays" model, which is the most common remuneration practice among CRAs, lends to business more than two thirds of their total revenues [256]. CRAs also publish unsolicited ratings [254], being considered less reliable and less accurate because they are based on publicly available data. Therefore, CRAs face a moral hazard problem, in which they have an incentive to overestimate the creditworthiness of the issuers and a restrain to avoid the loss of their credibility with the investors.

Besides the above issue, several studies pointed out that CRAs provide different rating for the same entity [259] and the markets react differently to rating changes made by each agency [257]. These disagreements are more frequent for sovereign ratings than for corporate ones [260], between one or two notches in the finer risk-scale [259], and may be explained by the use of varying economic and non-economic factors and different weights on these factors, as well as the different methodologies [261]. Even though CRAs publish their rating methodologies, the precise models are not officially disclosed because of their business practice. In addition, the qualitative part of the rating approach makes it harder to identify the relationship between the assessment criteria and the resulting sovereign ratings [255], aggravating the problem of opacity in the rating process.

In order to complement or replace human or institutions decisions, many statistical and machine learning techniques had been applied to financial and business issues [43, 262–264]. In this sense, the sovereign rating problem is a multiclass classification problem in which the items require to be classified into naturally ordered classes. However, even though the ordered nature of most of the financial classification problems, most of the solutions apply nominal classification techniques. This chapter deals with the problem of sovereign rating within an ordinal classification framework, and more precisely with the proposed PCDOC.

In addition to the classification task under the ordinal classification focus, we use the projection of the regressor model in PCDOC for ranking visualization which might be suitable to build decision support systems. The advantage of using this projection is that, in contrast to unsupervised visualization and projection techniques, here we have a projection which is validated by means of its suitability to correctly classify patterns. Figure 6.1 shows a graphical summary of the whole process described in this chapter.

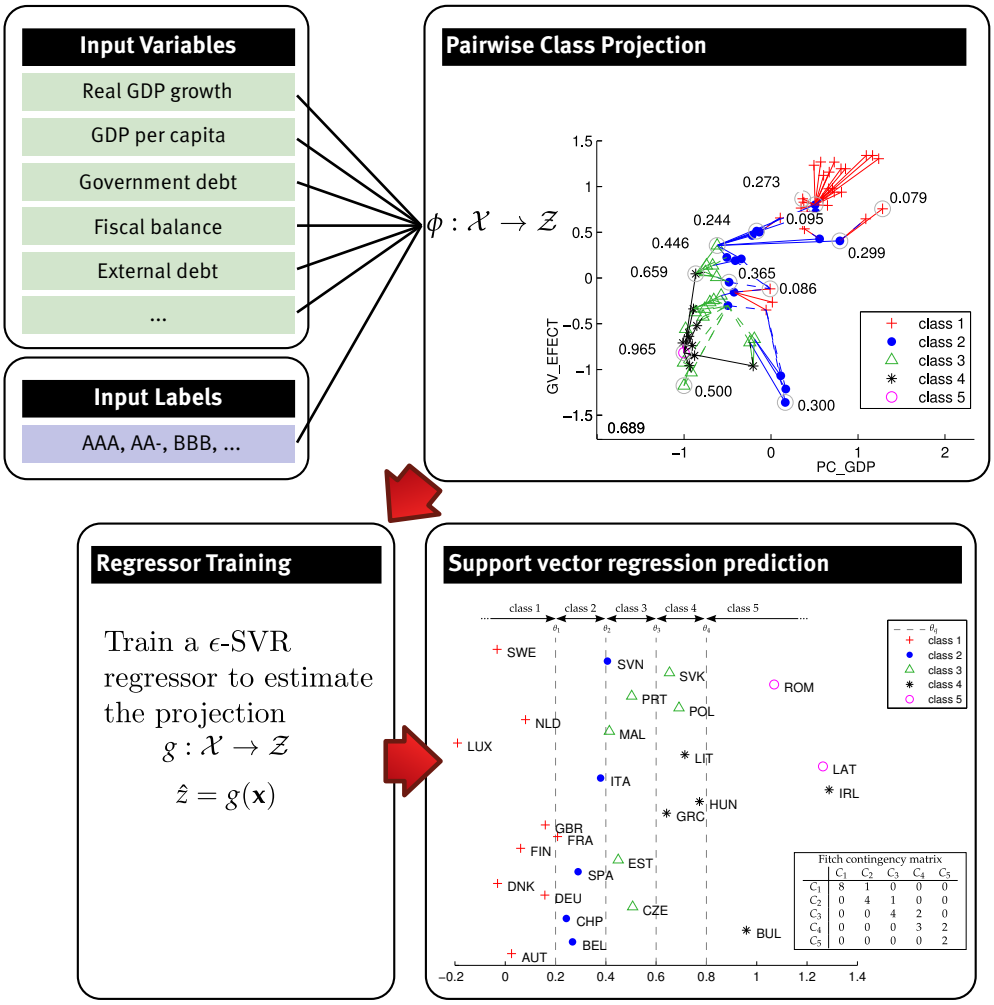


Figure 6.1: Graphical illustration of the work flow of PCDOC methodology with application to the sovereing ratings.

6.2 MACHINE LEARNING STATE-OF-THE-ART FOR SOVEREIGN RATING

This section briefly presents the related state-of-the-art works. An in-depth study is out of the scope of the current thesis, but there are several review papers regarding statistical and machine learning techniques applied to financial and business issues [43, 262–264].

In the accounting and finance domain, bankruptcy prediction and credit scoring are the two major research problems [43]. More related to the current work, sovereign debt rating issue is growing in attention of the machine learning scientific community,

although most of the methodologies have been focussing on corporate bonds rather than on sovereign risk [265]. In the second case, to assess the ability of a sovereign to honour its debt, some works applied statistical techniques such as Discriminant Analysis (DA) [266–268]. In spite of the easy to explain behaviour of the statistical models, the problem with applying these methods to the bond-rating prediction problem is that the multivariate normality assumptions for independent variables are frequently violated in financial data sets [269], which makes these methods theoretically invalid for finite samples [270]. This justify the use of alternative methods such as machine learning ones. The literature recognized the unsuitability of these approaches to deal with sovereign rating problem because they ignore the ordered nature of ratings and ordered response models have been later introduced to overcome this limitation [255, 271–274].

Machine learning methods have been applied to model sovereign ratings, e.g. Artificial Neural Networks (ANN), which do not rely on parametric assumptions of normality of data, independence of the explanatory variables, stationary or sample-path continuity. The better performance of ANN compared to previous statistical methods have been highlighted by Cosset and Roy [275], Cooper [276], Yim and Mitchell [277] and Benell [265], among others. Although this approach is not without its problems, such as the risk of over-fitting, the difficulty entailed in defining the physical structure of the network, and the tendency to fall into local optima [6]. Support Vector Machines [44, 58] have being widely used for financial problems in the recent years [43], for instance the standard SVM classifier has been applied to financial time series forecasting [278] or to corporate credit rating prediction [279]. Later, new SVM models have being evaluated for credit scoring, for example, weighted SVM models such as the Least Squares SVM (LSSVM), where the hyper-parameters selection and training are based on the Area Under receiver operating characteristics Curve (AUC) maximization [280]. Yu et. al presents a modified LSSVM to consider the prior knowledge that different classes may have different importance for model building so that more weight should be given to important classes [281]. Finally, soft computing techniques have been considered for financial problems. For instance, bank performance prediction were tacked with fuzzy SVM models by Chaudhuri and De [282] or with ensemble systems by Ravi et. al [283]. In addition, hybrid machine learning approaches have been applied to credit scoring [284] and bank rating [285].

However, most of the machine learning works deals with the problem as a binary classification problem, because several classifiers, such as SVM, are naturally designed for binary classification tasks. This would limit the applicability to evaluate credit as "risk" or "non-risk". More recent approaches not only use a multi-class focus, but a limited number of them also consider an ordinal perspective of the problem. Van Gestel et al. [143] propose a whole process model to develop rating systems. In this work the classifier side is implemented by adding SVM terms to the linear model of the ordinal logistic regression so that the final model is both accurate and readable. Kim and Ahn [6] studied different ensembles of nominal multi-class support vector machines (MSVM) and applied an ordinal pairwise partitioning to this MSVM to build an ordinal MSVM.

Table 6.1: A comparison of the rating labels from CRAs. The observations have been grouped into broader categories in the second column.

	Broader rating categories		S&P	Moody's	Fitch
Investment grade	Highest quality	C_1	AAA	Aaa	AAA
	High quality	C_2	AA+	Aa1	AA+
			AA	Aa2	AA
			AA-	Aa3	AA-
	Strong payment capacity	C_3	A+	A1	A+
			A	A2	A
			A-	A3	A-
	Adequate payment capacity	C_4	BBB+	Bbb1	BBB+
			BBB	Bbb2	BBB
			BBB-	Bbb3	BBB-
Speculative grade	Likely to fulfill obligations, ongoing uncertainty	C_5	BB+	Bb1	BB+
			BB	Bb2	BB
			BB-	Bb3	BB-
	High credit risk	C_6	B+	B1	B+
			B	B2	B
			B-	B3	B-
	Very high credit risk	C_7	CCC+	Caa1	CCC+
			CCC	Caa2	CCC
			CCC-	Caa3	CCC-
	Near default with possibility of recovery	C_8	CC	Ca	CC
			C	C	C
	Default	C_9	SD	-	SD
			D	-	D

6.3 EXPERIMENTS

This section presents the description of the sovereign rating dataset, the ordinal classification performance metrics, the dataset experimental design and related methods experiments configuration, and the comparison of these methods to the proposed one. The section concludes with the analysis of the predicted projection of the patterns for the generalization set, i.e. the 2010 year.

6.3.1 *Data description*

A set of economic descriptors of countries have been collected. The dataset contains 108 annual observations of long-term foreign-currency sovereign credit ratings of 27

Table 6.2: Description of the input variables. Note the business cycle approach has been considered through the inclusion of Gross Domestic Product (GDP) growth, fiscal and current account balance, inflation and unemployment as a three years average.

Variable name	Unit of measurement	Rating Influence
Real GDP growth	Rate	Positive
GDP per capita	Euros per inhabitant	Positive
Government debt	Percent of GDP	Negative
Fiscal balance	Percent of GDP	Positive
External debt	Percent of exports	Negative
Foreign reserves	Percent of imports	Positive
Current account balance	Percent of GDP	Negative
Inflation	Index (2005=100)	Negative
Unemployment	Rate	Negative
Unit labour cost	Index (2005=100)	Negative
Government effectiveness	Percentile	Positive

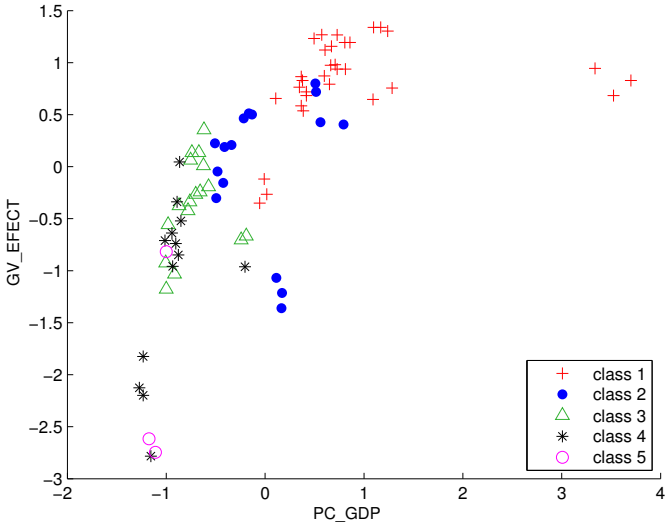
EU sovereign borrowers during the period from 2007 to 2010. The credit rating data is from the publicly available historical information provided by the three leader agencies, Standard and Poor's, Moody's and Fitch, as of December 31st of each year. The three CRAs use similar rating scales with 23-risk points (Table 6.1), in which the triple-A notation means the best quality issue. The observations have been grouped into broader categories according to Hill et al. [259], which are shown in the second column.

By using the broader rating categories of Table 6.1 the problem would be a 9 ordinal label classification problem. However, during the time span considered, all the European issuers were rated into the five first categories. Consequently, only five classes are considered for the classification problem (C_1, C_2, C_3, C_4, C_5).

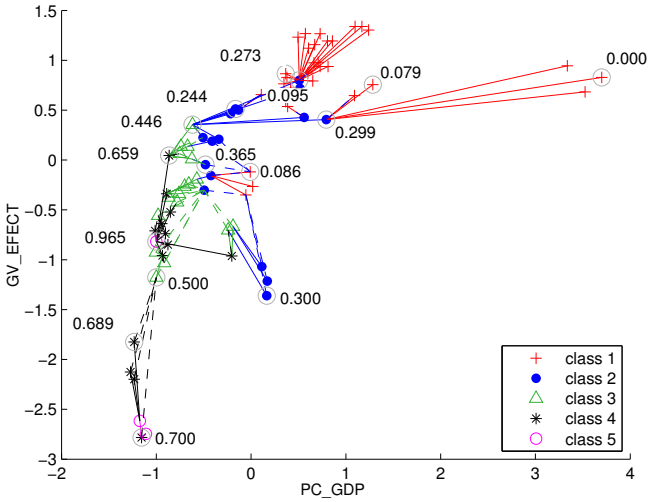
Many empirical studies, such as those from Cantor and Packer [260] and Bissoyondal-Beheninck [255], among others, have investigated the determinants of sovereign ratings, showing that they are mainly driven by economic variables. Based on them, eleven variables have been selected (see Table 6.2), ten economic indicators as well as one non-economic [286]. The data were taken from Eurostat, except Government Effectiveness indicator provided by the World Bank as well as the External Debt figures, which were completed with information provided by Central Banks of Cyprus, Malta, Sweden and Romania.

Notice that the methodology proposed attempts to estimate the rating provided by CRAs based on a reduced set of publicly available indicators in comparison with the large number criteria taken into account by CRAs. All the information about those criteria is accessible online into the CRA web pages, being comparable across the rating process, but they differ in the way in which they are classified and weighted by the firms' analysts [287, chap. III].

Three datasets are generated, one for each of the CRAs considered (Fitch, Moody's and S&P). The input variables are the same for all of them, but the rating is different



(a) Patterns of Fitch's dataset considering PC_GDP and GV_EFECT variables.



(b) Example of the generated z_i values on the Fitch's dataset.

Figure 6.2: Illustration of the PCD projections with PC_GDP and GV_EFECT variables of Fitch's dataset corresponding to patterns of years 2006-2009. Points of different classes are plotted with different symbols and colours.

depending on the CRA taken into account. The dataset for each CRA has been split in two subsequent time periods sets used as training and generalization sets. The first set includes 81 observations, described by the correspondent variables, from the 27 EU sovereign borrowers during the period 2007-2009, whereas, in the second one, data are from 2010.

6.3.2 *Analysis of the PCD projection with sovereign rating datasets*

Before continuing, we present this section as a preliminary analysis of the data, so that we can directly observe the exploitation of the order information done by PCD. For this purpose two different input variables have been selected from the sovereign rating datasets of Section 6.3.1. Our objective is to show how the proposed projec-

tion works in a two dimensional representation. The variables selected are "GDP per capita" (PC_GDP) and "Government effectiveness" GV_EFECT variables, for years 2006-2009, their values being standardised. This selection is done because these variables are suitable for showing the ordinal structure of the data in the input space, not because of feature selection criteria, which is out of the scope of this chapter. Fitch countries patterns labelling is used for this illustration example, as can be seen in Figure 6.2a. The figure shows that the data have a clear ordinal distribution through the input space, and how a separation in classes is difficult, some of them being clearly overlapped. It can be noticed the majority of patterns of each class are situated in regions of the space having adjacent classes patterns in neighbour regions.

Figure 6.2b presents the PCD concepts applied to the patterns in Figure 6.2a. The minimum distances are illustrated with lines of the same colour than the class. The minimum distance of a point to the next class patterns are marked with solid lines, while the minimum distances to the previous class are marked with dashed lines. For some example points (surrounded by a grey circle), the value of the PCD projection using Eq. (5.11) is shown near the point. It can be easily seen that the z value increases for patterns of the higher classes, and this value varies depending of the position of the pattern $\mathbf{x}^{(q)}$ in the space with respect to the patterns $\mathbf{x}^{(q-1)}$ and $\mathbf{x}^{(q+1)}$ of adjacent classes.

6.3.3 *Experimental design and comparison methods*

The experiments have being carried out by using a hold-out experimental design of the three datasets described at Section 6.3.1. The training dataset consist on patterns belonging to years 2006-2009. The generalization or test dataset consist on the 2010 year patterns. It must be paid attention to the Moody's dataset since during the period 2006-2009 (training period) there were not ratings of class C_5. However, in 2010 (test period), Greece was ranked as C_5 by this CRA. Given that all classifiers were trained for a four class dataset, none of them was able to correctly classify Greece during the test phase.

Due to the deterministic nature of all the compared methods, only one run of each method has been performed for each dataset and the generalization performance for several classification metrics is reported.

The ordinal regression methods used for comparison purposes have been selected according to their similarities to the proposal, and also because of the implementation availability. The ordinal methods considered are: A Simple Approach to Ordinal Regression (ASAOR) with a C4.5 base classifier (as suggested by Frank and Hall [27], ASAOR(C4.5)), Reduction from cost-sensitive ordinal ranking to weighted binary classification (RED) with SVM (RED-SVM) by Li and Lin [30], GPOR [41], SVOREX and SVORIM [33]. The reader can consult Section 3.2 for more details. In our approach, the Support Vector Regression (SVR) algorithm is used as the base regressor, so the method is called SVR-PCDOC. The ϵ-SVR implementation available in the libSVM package [169] is used. With respect to the ASAOR method, the C4.5 method available in Weka [66] is used as the underlying classification algorithm since this is the one initially employed by the authors of ASAOR. In this way, the algorithm is identified as ASAOR(C4.5).

In addition, some well known nominal algorithms have been selected in order to compare the ordinal approaches and to check if the ordinal classifiers are taking benefit from the order information. These are C4.5, a standard multinomial logistic regression (Mlogistic), a logistic regression based on simple regression and variable selection (Slogistic), and the MultiLayer Perceptron (MLP) neural network. A detailed description of these methods can be found in the works of Landwehr et al. [288] and Witten and Frank [117].

Model selection is an important issue and involves selecting the best hyper-parameter combination for all the methods compared. All the kernel methods were configured to use the Gaussian kernel. For the support vector algorithms, i.e. REDSVM, SVOREX, SVORIM and ϵ-SVR, the corresponding hyper-parameters (regularization parameter, C, and width of the Gaussian functions, γ), were adjusted using a grid search over each of the training set by a 5-fold cross-validation with the following ranges: $C \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$ and $\gamma \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$. Regarding ϵ-SVR, the additional ϵ parameter has to be adjusted. The range consider was $\epsilon \in \{10^{0}, 10^{1}, 10^{2}, 10^{3}\}$. GPOR has no hyper-parameters to fix, since the method optimizes the associated parameters itself. Finally, ASAOR(C4.5), C4.5, Mlogistic and Slogistic have not hyper-parameters. For the MLP method, the number of hidden neurons was also adjusted by cross-validation in the training set.

6.3.4 *Experimental results*

Table 6.3 presents the performance results of the different algorithms with the three datasets. Nominal and ordinal classifiers are separated with a horizontal line. In general, it should be pointed out that the performance ranking changes for each metric. However, SVR-PCDOC is very robust when considering all the datasets and all the metrics. It achieves the best performance for all the metrics in the Fitch and S&P datasets. In addition, the second best method for these datasets varies with respect to the metric considered. For example, Slogistic is the second best one in S&P when considering Acc. In this dataset, SVORIM was in the seventh position for Acc, but for metrics which consider the order (MAE, $AMAE$ and τ_b), this method is the second best one. Regarding Moody's dataset, SVR-PCDOC has the best results for MAE and τ_b, but not for Acc and $AMAE$. This is due to the error that the classifier has for Greece pattern in 2010, since it misclassifies Greece as C_3 when it a C_5 pattern, and this is more more penalized by $AMAE$ than by MAE.

Table 6.4 shows the credit rating granted by the three leader CRAs and the credit rating predicted by the SVR-PCDOC models for the 27 EU countries in the test set (year 2010). Errors have been highlighted in bold face. The data included in this table should be analysed together with the contingency matrices in Figures 6.3, 6.4 and 6.5.

If we take into account the test set (corresponding to year 2010), the number of patterns of each class are the following: $\{9, 5, 6, 5, 2\}$ for Fitch, $\{9, 5, 6, 6, 1\}$ for Moody's and $\{9, 3, 9, 3, 3\}$ for S&P. Considering the three datasets, the total distribution is $\{27, 13, 23, 14, 6\}$. Taking into account this distribution, a comparative analysis of per class classification error is now presented. The number of patterns in class C_1 is significantly higher than in other classes, particularly with respect to the class C_5. While the number of patterns misclassified is three of 27 in class C_1, it raised to three of six in class C_5. In the case of the intermediate classes, the misclassification errors are

Table 6.3: Comparison of the proposed method to other nominal and ordinal classification methods. The value of different metrics test results are reported for each dataset. The best result is in bold face and the second best result in italics.

	Accuracy			MAE		
Method/DataSet	Fitch	Moody's	S&P	Fitch	Moody's	S&P
C4.5	0.6296	0.6667	0.5926	0.4074	0.4074	0.4815
Mlogistic	0.4815	0.7778	0.3704	0.8889	0.3333	0.8889
MLP	0.6667	**0.8519**	0.6667	0.4074	0.2593	0.4444
Slogistic	*0.7407*	0.7778	*0.7037*	*0.2593*	0.2963	0.4074
ASAOR(C4.5)	0.5926	0.6296	*0.7037*	0.4815	0.4815	0.4074
RED-SVM	0.6667	*0.8148*	0.6667	0.3333	**0.2222**	0.4074
GPOR	*0.7407*	0.7037	0.6667	0.3704	0.4444	0.4444
SVOREX	0.7037	0.7778	0.5926	0.2963	0.2593	0.4444
SVORIM	0.6667	*0.8148*	0.6296	0.3333	**0.2222**	*0.3704*
SVR-PCDOC	**0.7778**	*0.8148*	**0.7407**	**0.2222**	**0.2222**	**0.2593**
	AMAE			τ_b		
Method/DataSet	Fitch	Moody's	S&P	Fitch	Moody's	S&P
C4.5	0.4400	0.6800	0.5111	0.7621	0.7367	0.7655
Mlogistic	1.1600	0.6467	0.9333	0.5255	0.7719	0.5121
MLP	0.5267	**0.4067**	0.4000	0.7972	0.8097	0.7492
Slogistic	0.2667	0.6200	0.5111	*0.8951*	0.8151	0.8060
ASAOR(C4.5)	0.4533	0.7533	0.4222	0.6989	0.6655	0.7570
RED-SVM	0.2822	*0.5356*	0.4222	0.8835	0.8590	0.8052
GPOR	0.5133	0.9200	0.6222	0.7738	0.6869	0.7807
SVOREX	*0.2422*	0.5622	0.4444	0.8886	**0.8610**	0.7873
SVORIM	0.2756	*0.5356*	*0.3556*	0.8799	0.8525	*0.8370*
SVR-PCDOC	**0.2089**	0.5467	**0.2889**	**0.9224**	**0.8610**	**0.8849**

lower for class C_4, only two errors, compared to classes C_2 and C_3, with four and six errors, respectively.

The models could classify patterns in an upper or a lower class than that rated by the CRA. In our analysis, we will use the terms "positive error" to mean that model classifies the pattern in upper classes and "negative error" for lower classes compared to the real ones (the higher the class the worse the rating is for the country). The results show that the model for CRA Fitch committed 6 positive errors across the fourth first classes, while the models for the other two CRA did both type of errors. The errors committed by Moody's model were less biased because it misclassified positively two patterns and negatively other three, being located into the classes C_2, C_3 and C_5. Finally, the predicted label in the S&P model is very close to real rating. This model committed 7 errors, 4 positive and 3 negative, across all the classes unless C_4. In all cases, the model misclassified patterns in adjacent classes to the real one.

Table 6.4: Ratings for EU countries in the test set (2010): real ratings versus SVR-PCDOC predicted rating.

		Fitch		Moody's		S&P	
Pattern	Acronym	Rating	Pred.	Rating	Pred.	Rating	Pred.
Austria	AUT	1	1	1	1	1	1
Belgium	BEL	2	2	2	2	2	2
Bulgaria	BUL	**4**	**5**	4	4	4	4
Cyprus	CHP	2	2	2	2	**3**	**2**
Czech Republic	CZE	3	3	3	3	3	3
Germany	DEU	1	1	1	1	1	1
Denmark	DNK	1	1	1	1	1	1
Estonia	EST	3	3	3	3	3	3
Finland	FIN	1	1	1	1	1	1
France	FRA	**1**	**2**	1	1	**1**	**2**
Great Britain	GBR	1	1	1	1	**1**	**2**
Greece	GRC	4	4	**5**	**3**	**5**	**4**
Hungary	HUN	4	4	4	4	4	4
Ireland	IRL	**4**	**5**	4	4	3	3
Italy	ITA	2	2	2	2	3	3
Latvia	LAT	5	5	4	4	**5**	**4**
Lithuania	LIT	4	4	4	4	4	4
Luxembourg	LUX	1	1	1	1	1	1
Malta	MAL	3	3	3	3	3	3
Netherlands	NLD	1	1	1	1	1	1
Poland	POL	**3**	**4**	3	3	**3**	**4**
Portugal	PRT	3	3	**3**	**2**	3	3
Romania	ROM	5	5	4	4	5	5
Spain	SPA	2	2	**2**	**1**	**2**	**3**
Slovakia	SVK	**3**	**4**	**3**	**4**	3	3
Slovenia	SVN	**2**	**3**	**2**	**3**	2	2
Sweden	SWE	1	1	1	1	1	1

Errors have been highlighted in bold face.

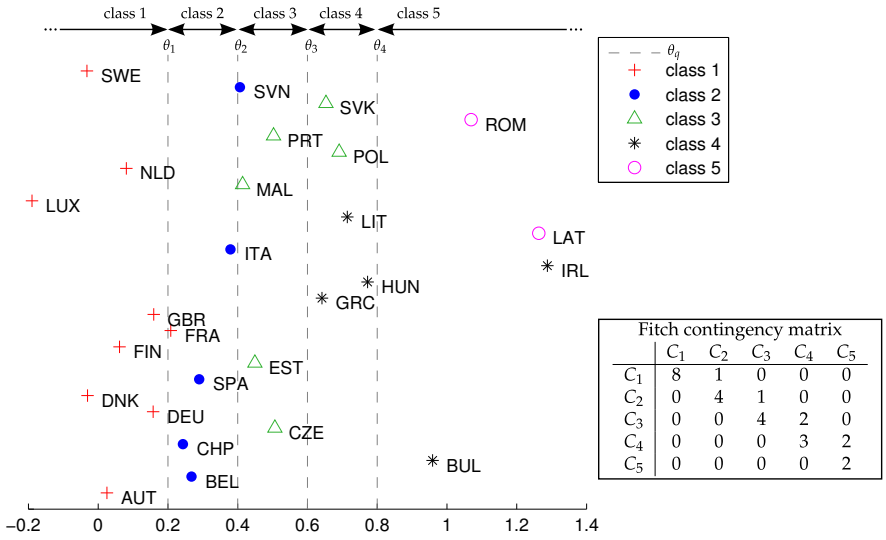
We can distinguish several groups among the misclassified patterns. The first group encompassed the countries that have recently joined to the European Union (Bulgaria, Slovakia, Slovenia and Poland). For these countries, their EU membership represents a qualitative feature that is positively valued throughout the credit risk assessment [289]. The non-inclusion of this aspect in the set of the explanatory variables could partly explain the negative errors committed by the models with respect to these countries.

The second group encloses two great European powers, France and Great Britain. The economic downturn had led distortions in some of their economic fundamentals, bringing them closer to those countries characterized as class C_2. However, their economic, political and financial structure generate favourable short and medium-

term expectations concerning to their creditworthiness, which allow them to gain the highest rating for their sovereign debt. The third group comprises the PIGS countries, strongly punished by financial markets during the sovereign debt crisis episode. Portugal, Spain and Greece are negatively misclassified by some models. In this case, the differences between projected and real rating is due to the negative expectations on their future economic performance, especially those aspect related to the fiscal policy. As for Greece, the reliability of the data employed [290] may also be the cause of its better performance. On the other hand, Fitch model committed a positive error classifying Ireland. This error could be due to its higher fiscal deficit compared to those patterns in classes C_4 and C_5.

Finally, the case to be analysed is that of Cyprus which is assigned by the three models to class C_2. Thus, the S&P model misclassified negatively this pattern. It also seems to indicate that S&P model did not reflect the fact that S&P tends to downgrade issuers when compared to the other two agencies [261]. Indeed, S&P downgraded Cyprus from C_2 to C_3 at the end of 2010, while Moody's and Fitch did it during the first half of the following year.

6.3.5 *Analysis of the predicted projection*



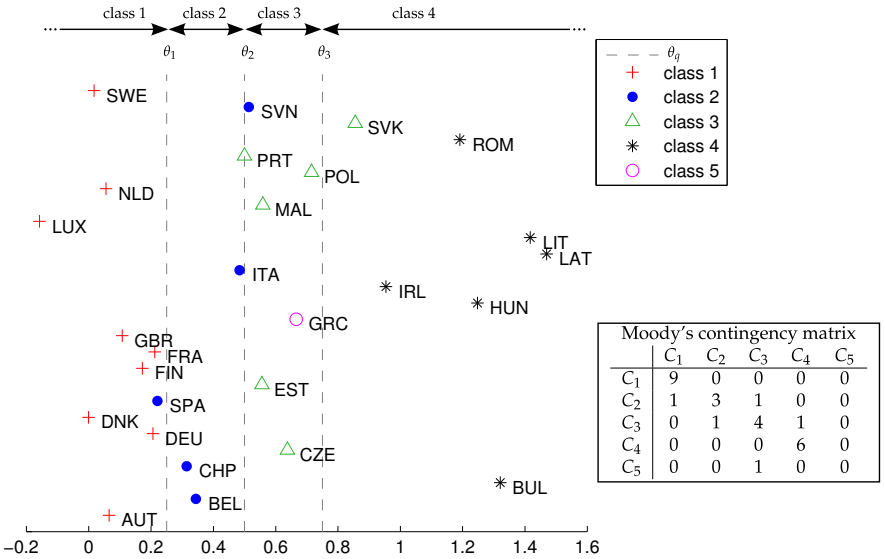
Fitch contingency matrix

	C_1	C_2	C_3	C_4	C_5
C_1	8	1	0	0	0
C_2	0	4	1	0	0
C_3	0	0	4	2	0
C_4	0	0	0	3	2
C_5	0	0	0	0	2

Figure 6.3: Predicted projection values and contingency matrix for the Fitch test set (year 2010) by using SVR-PCDOC.

This section analyses the values predicted by the regressor in the latent space ($\mathcal{L}$) for the three datasets, which are generated by the SVR-PCDOC model. Due the way the PCD projection is built, and the quality of the regressor model –which is validated through its classification performance–, it makes sense to use these predictions to provide additional information to potential decision makers. We propose to use these predicted values as a one dimensional measure of the overall patterns rank.

Figures 6.3, 6.4 and 6.5 show the predicted one dimensional projection of the SVR-PCDOC model. This is, the value predicted by the SVR model trained with the PCD projection. Patterns of different classes are highlighted with different colours and symbols corresponding to the actual class. Thresholds for each class are plotted with



Moody's contingency matrix

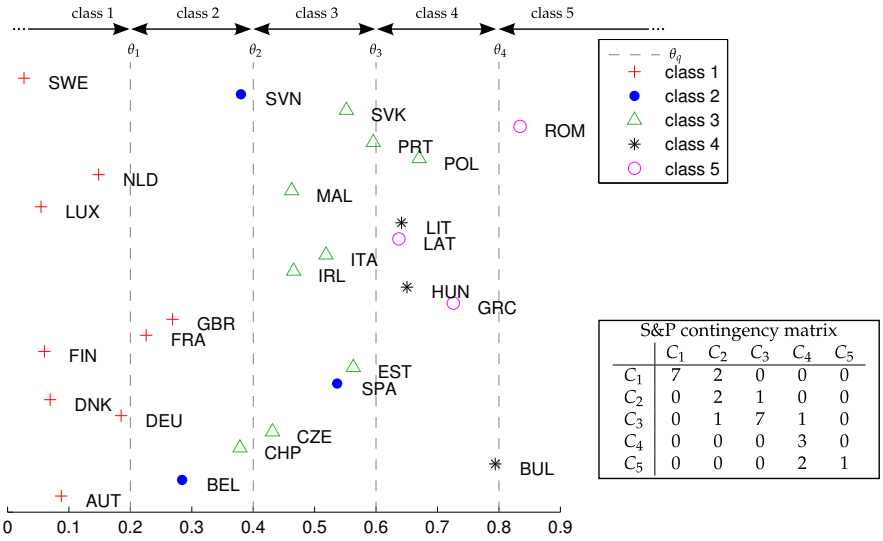
	C_1	C_2	C_3	C_4	C_5
C_1	9	0	0	0	0
C_2	1	3	1	0	0
C_3	0	1	4	1	0
C_4	0	0	0	6	0
C_5	0	0	1	0	0

Figure 6.4: Predicted projection values and contingency matrix for the Moody's test set (year 2010) by using SVR-PCDOC.

dashed lines so that it can be seen whether the prediction matches the right class or not (a pattern is classified into a class depending on how it is relatively positioned with respect to the thresholds, see Eq. (3.1)). Observe that for Moody's (Figure 6.4) there are only three thresholds since the training set contained patterns for only four classes.

For all the plots, the position of the patterns into the one dimensional variable space should be taken into account, so that patterns close to the thresholds are more likely to be misclassified, and it would be advisable to get an expert revision of the final classification to complete a more robust decision support system. This is the case of Great Britain and France rated with C_1 for the Fitch and Moody's datasets (see Figures 6.3 and 6.4). However, the predictions of the models are very close to the thresholds, and these predictions are quite consistent with the ones of the S&P model that definitely places Great Britain and France in C_2. On the other hand, there are some countries that are placed in the minimum values of the first class (C_1) interval (e.g. Luxembourg, Sweden, Finland, Denmark or Austria). According to the input variables, the trained model places this countries as "better positioned" on class one than all the other patterns.

Together with the plots, it is included the corresponding confusion or contingency matrix of the classification results. Observe that errors are aligned through the diagonal of the matrices (as mentioned during this dissertation, e.g. in Section 3.2, this is one of the aims of the ordinal classification) The only exception is the case of Greece in Moody's dataset (see Figure 6.4), where this pattern of C_5 is placed on class C_3. However, we can argue that this case is not common since the model were trained by only considering four classes.



S&P contingency matrix

	C_1	C_2	C_3	C_4	C_5
C_1	7	2	0	0	0
C_2	0	2	1	0	0
C_3	0	1	7	1	0
C_4	0	0	0	3	0
C_5	0	0	0	2	1

Figure 6.5: Predicted projection values and contingency matrix for the S&P test set (year 2010) by using SVR-PCDOC.

6.4 CONCLUSIONS

In this chapter, we have applied the PCD projection as a suitable methodology for data classification and validation. The robustness of the classifier is demonstrated using four performance metrics for comparing the PCDOC classifier to nominal and ordinal classifiers in the three main CRA's datasets. In most cases, the errors committed by the three models implies the misclassification of patterns in only one upper or lower class rather than several ranks in the ordinal rating scale. It supposes an advantage for decision making process based on scenarios considering sovereign ratings.

On the other hand, the pattern by pattern analysis indicates that the set of explanatory variables has to be augmented with other qualitative variables for some countries. In this regard, the historical information about a country's economic performance could be completed with data on economic short and medium-term expectations, as a result projected rating would turn into forward-looking evaluation of the country's ability to honour its sovereign debt.

This study can be extended in two different ways to increase the accuracy of the proposed models, once data comparable across countries and for periods of less than one year are available. First, by using quarterly or monthly data that provide more detailed information on the economic and financial developments in the context of the high volatility and the repricing of risk in financial markets. Secondly, by including data about country banking sector's exposures to sovereign debt and housing bubble that finally crystallize on the government's balance sheet.

7

APPLICATION OF ORDINAL REGRESSION TO WIND SPEED FORECASTING

Summary. This chapter addresses the problem of wind speed forecasting under the umbrella of ordinal regression.

In general, previous works consider the wind speed as a continuous target variable, estimating then the corresponding wind series of continuous values. However, the exact wind speed is not always needed by wind farm managers, and a general idea of the level of speed is, in the majority of cases, enough to set functional operations for the farm (such as wind turbines stop, for example). Moreover, the accuracy of the models obtained is usually improved for the classification task, given that the problem is simplified.

Therefore, this chapter tackles the problem of wind speed prediction by considering wind speed as a discrete variable and, consequently, wind speed prediction as a classification problem, with four wind level categories which a clear order arrangement: low, moderate, high or very high.

Associated publications. Some portions of this chapter appeared in [5, 291]:

- P.A. Gutiérrez, S. Salcedo-Sanz, C. Hervás-Martínez, L. Carro-Calvo, J. Sánchez-Monedero, and L. Prieto. Ordinal and nominal classification of wind speed from synoptic pressure patterns. *Engineering Applications of Artificial Intelligence*, 26(3):1008–1015, 2013.
  JCR: 1.665
  http://dx.doi.org/10.1016/j.engappai.2012.10.018
- P.A. Gutiérrez, S. Salcedo-Sanz, C. Hervás-Martínez, L. Carro-Calvo, J. Sánchez-Monedero, and L. Prieto. Evaluating nominal and ordinal classifiers for wind speed prediction from synoptic pressure patterns. In *Proceedings of the 11th International Conference on Intelligent Systems Design andApplications (ISDA 2011)*, pages 1265–1270, Cordoba, Spain, Spain, nov 2011.
  http://dx.doi.org/10.1109/ISDA.2011.6121833

7.1 INTRODUCTION

Among renewable energies, wind power is one of the most promising sources of renewable energy in the world, and also the one with a stronger economic impact in

developed countries [292]. As an example, wind power installed worldwide by the end of 2009 reaches a total of 157 GW, of which about 76 GW correspond to Europe, and 19 GW only to Spain. Thus, wind power represents over 12% of the total energy consumed in countries such as USA, Germany or Spain, and it is expected that this percentage grows up to an amazing 20% by 2025 [293]. This booming of wind energy has brought together the construction of a huge number of wind farms in the last few years, and, consequently, a good number of new problems associated with the management of these facilities.

Wind speed reconstruction, long-term prediction and wind series analysis are mainly the most important problems faced by wind farm managers in daily operations. These problems are related to different important decisions about the wind farm, such as maintenance stops, production analysis and planning and even micro sitting of new wind turbines. Existing approaches for these problems are mainly based on historic registers of wind measures, from which statistical models are constructed in order to explain the wind behaviour. These models can be then applied to future values of time in the case of long-term wind speed prediction, or to values in the past in order to reconstruct or analyse and reconstruct wind speed series. Different techniques have been used to obtain these wind speed models, such as statistical methods [294, 295], neural networks [296, 297], Support Vector Machines [298], Bayesian models [299], etc. The majority of the existing techniques used to construct long-term wind speed models are exclusively based on past wind speed data, and some of them include other atmospheric variables as input data, such as local temperature, radiation or pressure at the measuring point. The problem with this approach based on wind measures is that, in some cases, these data are not available, due to fails in the measurement systems, or just because the terrain is a prospective site to install a wind farm, and there is not a meteorological tower installed yet. This problem is even harder in the case of historic analysis or wind series reconstruction, since it is not possible to obtain any direct wind measure if it is not available.

In these problematic cases, the possibility of obtaining indirect measures of wind is currently a hot topic, in which many renewable energy companies are investing lots of resources. In this sense, different recent works have used synoptic pressure[1] as an indirect measure to study different atmospheric phenomenons such as precipitation, pollution or temperature [300–306]. In the case of the wind, it seems even more evident that a good source of indirect wind measures is the pressure at synoptic scale, since the wind at a given point is a direct function (when the effects of limit boundary layer are removed) of the pressure gradient. Thus, different works have related pressure patterns with local or mesoscale wind [307–311]. Among them, the work by Hocaoglu et al. [310] has been selected in the experimental section as one of the compared methods, given that it resembles the proposal in this chapter in some ways.

Specifically, in this chapter, the problem of wind speed estimation in a given point (wind farm), from the corresponding synoptic pressure pattern is tackled. The problem involves daily pressure patterns in a synoptic grid, in this case centred in Spain, and a wind speed module measure. The main novelty of the chapter is that this

1 Synoptic scale in meteorology corresponds to atmospheric phenomenons in a horizontal length scale of the order of 1000 kilometres or more. Regarding to synoptic pressure, the majority of high and low-pressure areas that can be seen on weather maps are synoptic-scale systems.

wind speed is discretized into different levels of wind (classes) in order to treat it as a classification problem. The motivation behind this is that the manager of the wind farm can get enough information from the considered classes in order to set functional operations for the farm (such as wind turbines stop, for example). Note that the exact wind speed value is not usually important for this task. Additionally, higher accuracy can be obtained for a classification task, given that the problem is simplified. Four classes have been considered that cover all the wind speed spectrum of a wind farm operation.

7.2 PROBLEM DEFINITION

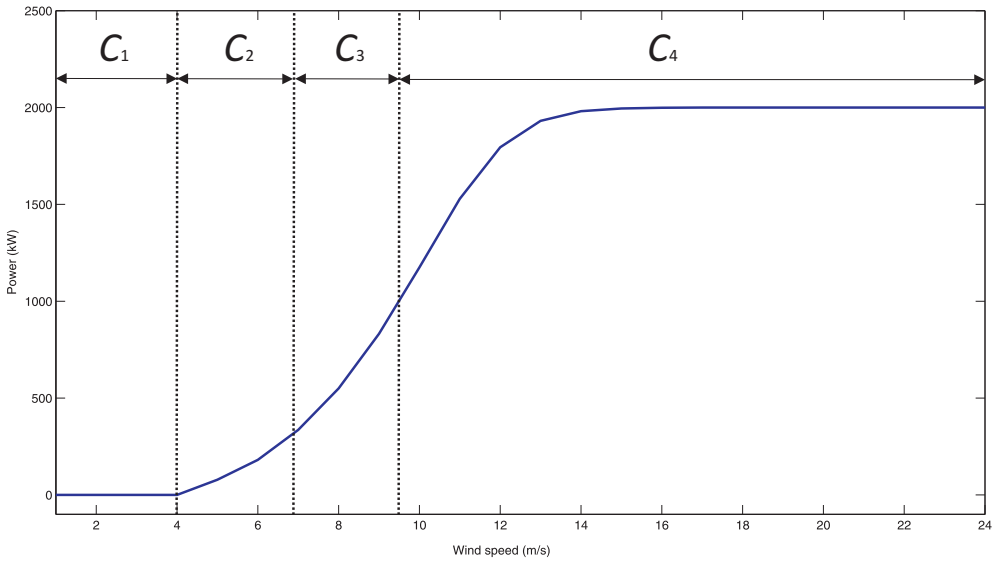


Figure 7.1: Wind speed classes ($C_1 \prec C_2 \prec C_3 \prec C_4$) and its relationship with the power curve of the wind turbines.

Formally we can define the wind forecasting problem as follows: Let $\mathbf{y} = \{y_i, i = 1, \ldots, N\}$ be a series of daily wind speed discretized measures at a given point, in such a way that $y_i \in \mathcal{Y} = \{C_1, C_2, C_3, C_4\}$, i.e. y belongs to one out of 4 classes which are subjected to an order ($C_1 \prec C_2 \prec C_3 \prec C_4$, where $\prec$ is an ordering relationship between the labels).

In this proposal, the different classes for the wind speed have been constructed taking into account the characteristics of the wind turbines, i.e. its power curve. Figure 7.1 shows the 4 classes established in this chapter, which try to model the power curve of the turbines installed in the considered wind farms. Thus, C_1 contains situations of low wind, where the wind turbine will not produce power, C_2 summarizes situations in the beginning of the wind power ramp, C_3 comprises situations in which the production of the wind turbine is significant and C_4 models situations of high wind speed and power production. Note daily averages of wind speed are studied, so the classes are set to have enough number of samples from each class. Let $\mathbf{X} = \{\mathbf{x}_i, i = 1, \ldots, N\}$ be a series of daily synoptic-scale pressure measures in a grid. In this case, each component of $\mathbf{X}$ is a matrix of 14×13 surface pressure val-

ues (182 values), measured in a grid surrounding the Iberian Peninsula (Figure 7.2). The problem faced in this chapter is a classification problem, consisting of obtaining a machine Φ by using a training set $\{(\mathbf{x}_i, y_i), i = 1, \ldots, N_n < N\}$ (the first part of the series), so that for a given value of $\mathbf{x}_i$, it estimates the associated value of y_i, i.e. $\Phi(\mathbf{x}_i) \rightarrow y_i$, in such a way that the machine Φ minimizes an error measure in an independent test set $\{(\mathbf{x}_i, y_i), i = N_n + 1, \ldots, N\}$ (the rest of the series), to ensure the good generalization of the machine.

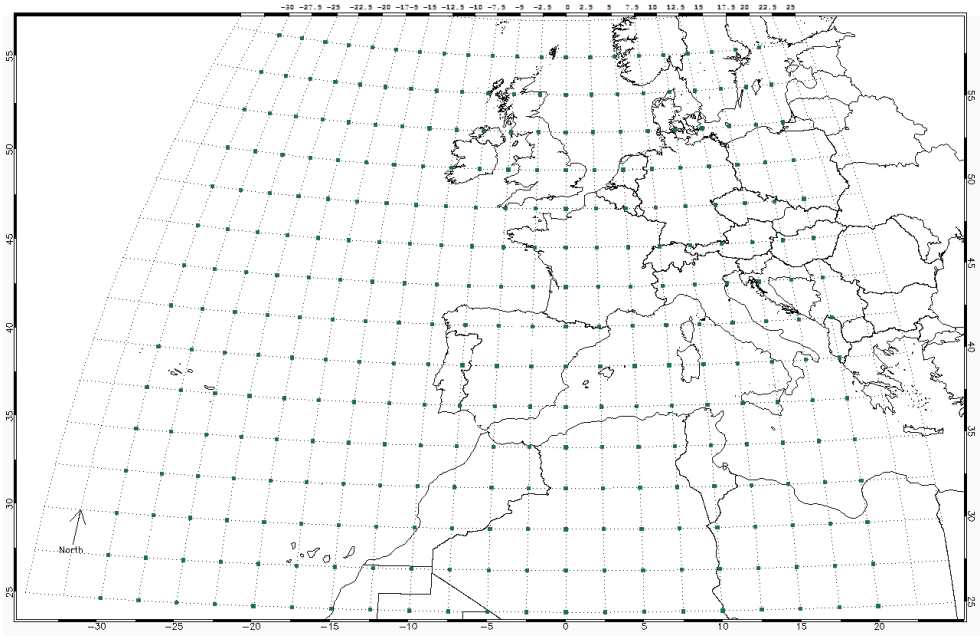


Figure 7.2: Synoptic pressure grid considered (Sea Level Pressure values have been used in this chapter).

7.3 EXPERIMENTS

In the following subsections, the description of the datasets and the experimental design is given, together with the description of the methods based on HMMs, which will be used also for comparison purposes. Then, the details on the preprocessing of the datasets are explained, and finally the obtained results with the different considered classifiers are discussed.

7.3.1 *Data description and preprocessing*

Five different wind farms have been considered for this study, resulting in five datasets (H, M, P, U and Z, see Figure 7.3). Each dataset includes a series of discretized wind speed values (targets), taken in a tower at 40m of height, and averaged over 24 hours to obtain daily data values. On the other hand, a series of grids of average daily pressure maps for the same period have been obtained from the National Center for Environmental Prediction/National Center for Atmospheric Rearch Reanalysis Project (NCEP/NCAR) [312, 313], which are public data profusely used

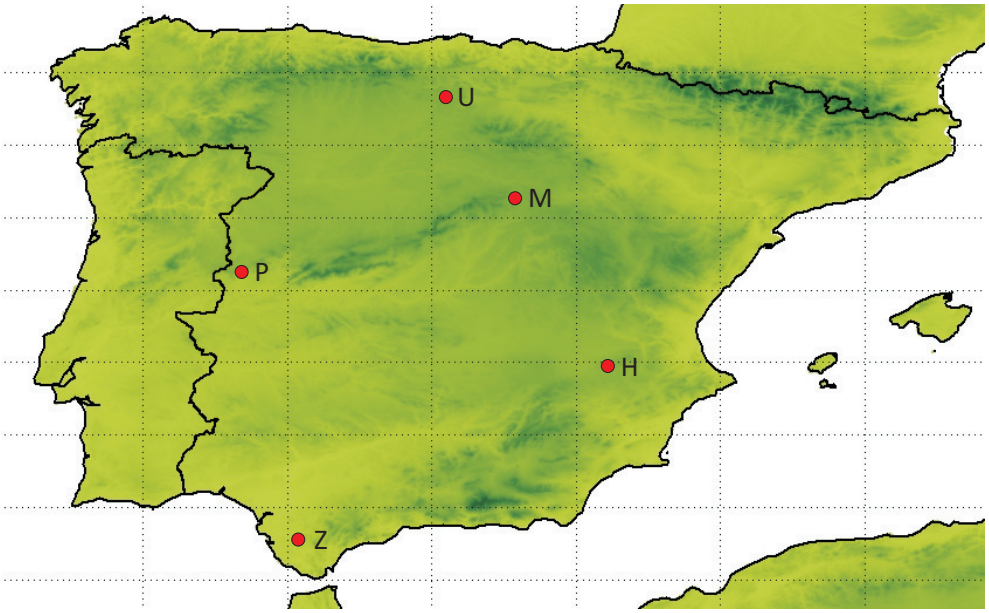


Figure 7.3: Location of the wind farms considered in this work.

Table 7.1: Structure of training and test sets: total number of patterns (Size), number of pattern in each class (C_1, C_2, C_3, C_4) (Distribution) and final number of Principal Components (PCs)

Wind	Training		Test		
farm	Size	Distribution	Size	Distribution	PCs[1]
H	2196	(416,1478,272,30)	1098	(200,790,99,9)	13
M	2231	(220,1590,396,52)	1115	(173,779,147,16)	10
P	2185	(773,1076,295,41)	1092	(409,538,125,20)	11
U	2017	(527,1167,280,43)	1008	(361,547,85,15)	6
Z	1749	(901,637,184,27)	874	(516,279,68,11)	13

[1] This value has been obtained using the algorithm in Figure 7.4.

in climatology and meteorology applications. As previously mentioned, an uniform grid in latitude and longitude has been considered, shown in Figure 7.2, with 182 measurement points, and each element of this grid is one input variable.

For each wind farm, two different sets are obtained, one for training the models and another one for assessing the performance of the algorithms. In this way, the structure of the different datasets used in this study is given in Table 7.1. The structures of these datasets are challenging, because the distribution of the different classes is clearly imbalanced, with very few situations of high wind speed (class C_4) and lot of patterns belonging to a moderate wind speed class (class C_2).

Since all the tested algorithms are deterministic, they will be run once, deriving a model from the training set and evaluating its accuracy over the test set. Both training

Deciding # of Principal Components::
Require: Training dataset (Tr), Test dataset (Te)
Ensure: Projected training dataset (Tr^*), Projected test dataset (Te^*)
1: Apply PCA to Tr, without considering Te
2: $Max \leftarrow$ Number of PCs retaining a 99% of the total variance of the dataset
3: **for** $i = 1 \rightarrow Max$ **do**
4: $Tr_i \leftarrow Tr$ projected over the i first PCs.
5: Apply a ten-fold cross-validation method, considering Tr_i data and the LDA classifier.
6: $e_i \leftarrow$ cross-validated error of the classifier.
7: **end for**
8: $n \leftarrow \text{argmin}_i e_i$
9: $Tr^* \leftarrow Tr$ projected over the n first PCs.
10: $Te^* \leftarrow Te$ projected over the n first PCs.
11: **return** Tr^* and Te^*

Figure 7.4: Algorithm for deciding the number of principal components.

and test sets are parts of a wind series, so it is not advisable to do different random partitions of them.

For the selection of the SVM's hyper-parameters (regularization parameter, C, and width of the Gaussian functions, γ), a grid search algorithm was applied with a ten-fold cross-validation, using the following ranges: $C \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$ and $\gamma \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$. This cross-validation has been applied only taking into account the training data, and then repeating the process with the lowest error parameter combination using the complete training set.

7.3.2 *Preprocessing of the dataset*

As previously stated, the vector of inputs is formed by 14×13 surface pressure values (182) values in a grid around the Iberian Peninsula), which results in a very high number of variables. When too many inputs are presented to the standard machine learning algorithms, a very well known problem appears, the *curse of dimensionality*, which can decrease the performance of these algorithms and significantly increase the computational cost. This is not needed for the HMMs described in subsection 7.3.4, given that only one single pressure value is obtained from the grid to construct the model.

In order to alleviate this problem, a simple approach has been applied, based on the standard technique of Principal Component Analysis (PCA) [314]. PCA is the predominant linear dimensionality reduction technique, and has been widely applied to datasets in all scientific domains. Generally speaking, PCA maps data points from a high dimensional space to a low dimensional space, while keeping all the relevant linear structure intact.

PCA algorithm returns so many principal components (PCs, linear combinations of the input variables) as the total number of inputs, but they are sorted in the following way: the first PC has as high variance as possible (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has

the highest variance possible under the constraint that it will be orthogonal to (i.e. uncorrelated with) the preceding components. Note that it should be decided at a later stage how many PCs are retained when reducing the dimensionality of the problem.

With this aim, the algorithm included in Figure 7.4 has been applied. The idea is very simple: the coefficients of the PCs are obtained using the training data and all possible combinations from 1 to the number of PCs that retain a 99% of the variance are tested. A 10-fold cross-validation is applied for each combination, estimating the error with one of the simplest existing classifier (a Linear Discriminant Analysis, LDA) in order to limit the computational time. Once the best number of PCs is decided, training and test data are projected into them, and the reduced datasets are returned.

7.3.3 *Comparison methods*

7.3.3.1 *Ordinal classifiers*

The ordinal regression methods used for comparison purposes are the following:

- ASAOR with a C4.5 base classifier (as suggested by Frank and Hall [27].
- RED-SVM by Li and Lin [30].
- SVOREX and SVORIM [33].
- GPOR [41].

The reader can consult Chapter 3 for more details about the ordinal methods.

7.3.3.2 *Nominal classifiers*

Very well-known standard nominal classifiers have been taken into account. Their main characteristics are briefly described in the following subsections.

Apart from the well-known Support Vector Machines, other standard machine learning classifiers have been considered. This set of classifiers have shown to report good performance in previous machine learning works [288], and they have been selected because they cover some of the more common and accurate approaches for nominal classification (classification trees, boosting ensemble construction, and logistic regression) from those available in the well-known Weka machine learning software [66]. They include:

- The Logistic Model Tree (LMT) classifier [288].
- The C4.5 classification tree inducer [315].
- The AdaBoost.M1 algorithm, using C4.5 as the base learner. AdaBoost, short for Adaptive Boosting, is a machine learning meta-algorithm [316], an algorithm for constructing a "strong" classifier as linear combination of simple "weak" classifiers. The maximum number of iterations has been set to 10 and 100 iterations (Ada10 and Ada100), as done in previous studies [288].

- Multi-logistic regression methods, including the MultiLogistic (MLogistic) and SimpleLogistic (SLogistic) algorithms.
  - MLogistic is an algorithm for building a multinomial logistic regression model, which is one of the more popular approaches for classification. The algorithm includes a ridge estimator to regularize the model and guard against over-fitting [317]. The coefficient matrices is found by a Quasi-Newton Method: the active-sets' method with the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update.
  - SLogistic is an alternative algorithm to build a multinomial logistic regression model. The process involves using the LogitBoost algorithm [318] to fit *additive logistic regression models* by maximum likelihood. These models are a generalization of the (linear) logistic regression models. This version of the algorithm is based on controlling the number of variables of the model to avoid over-fitting [288]: an iterative process adds the input variables one by one, and the number of iterations is decided using a cross-validation process.

7.3.4 *Comparison to Hidden Markov Models*

Apart from the methods presented, the approach of Hocaoglu et al. [310] based on Hidden Markov Models (HMMs) has been also selected. Although our work has some similarities with the approach presented in this paper (given that wind speed is also estimated from pressure data), some differences have to be outlined. First of all, a complete synoptic grid, with 182 different values (14×13) is considered in our approach (see Figure 7.2). However, the aforementioned paper considered one single atmospheric pressure observation. Pressure and wind speeds values are then quantized in different number of intervals in order to apply discrete HMMs to estimate wind speed. Other important fact is that in [310] hourly wind speed prediction is considered, whereas in the current approach we manage average daily wind speed values. The lower variability of these daily values can make necessary to use a lower number of states for modelling.

To adapt the approach in [310] to the proposal of this work, a HMM for each wind farm was constructed, considering one single-point pressure value obtained from the 182 values of the grid. Specifically, the absolute value differences between the upper left and the upper right points and between the bottom left and the bottom right ones were averaged. The number of states of each HMM was fixed to 4 states, considering the intervals for wind speed in Figure 7.1. The observable emissions were considered to be the single-point pressure values, which were discretized in 150 values (in a similar way to [310]). The transition probabilities are obtained and organized in a matrix form in the same way than in [310], as well as the emission matrix.

7.3.5 *Results*

The results for the two different evaluation measures considered (C and MAE see Section 3.2.4 for metrics definition) are included in Tables 7.2 and 7.3, respectively. Based on the C and MAE values, the ranking of each method in each wind farm is

Table 7.2: Test accuracy ($C(\%)$) results obtained by using the different methods evaluated.

	Wind farm						
Classifier	H	M	P	U	Z	$\overline{C}(\%)$	$\overline{R}_C$
SVM	**75.77**	*73.32*	63.64	62.50	**70.48**	69.14	3.10
LMT	75.05	70.94	56.23	62.80	64.65	65.93	6.10
C45	67.67	70.04	50.37	57.54	55.72	60.27	10.60
Ada10(C45)	69.22	68.43	53.66	**63.39**	61.10	63.16	8.40
Ada100(C45)	74.23	72.20	60.07	62.50	66.36	67.07	5.80
MLogistic	74.50	71.66	52.75	57.44	62.24	63.72	8.00
SLogistic	75.05	71.21	54.03	57.34	62.36	64.00	7.60
ASAOR(C45)	70.86	70.76	56.41	57.34	56.75	62.42	9.30
RED-SVM	**75.77**	**73.90**	63.64	62.50	**70.48**	**69.26**	**2.70**
SVOREX	*75.50*	73.27	**64.65**	*62.90*	68.99	69.06	*2.90*
SVORIM	75.23	73.36	*64.56*	*62.90*	*69.34*	*69.08*	**2.70**
GPOR	71.95	69.87	49.27	54.27	59.04	60.88	10.80
HMM	71.13	69.42	42.49	51.39	56.29	58.14	12.00

The best result is in bold face and the second best result in italics

obtained ($R = 1$ for the best performing method and $R = 12$ for the worst one). The mean accuracy and MAE ($\overline{C}$ and $\overline{M}$) as well as the mean ranking ($\overline{R}_C$ and $\overline{R}_M$) are also included in Tables 7.2 and 7.3 ($\overline{R} = 1$ for the best method and $\overline{R} = 13$ for the worst one). The first conclusion is that considerably good accuracies are obtained, what reveals that considering the problem as a classification task can provide an accurate information of the wind farm. Also, the MAE values are quite low, the algorithms doing a quite good job when ranking the patterns (a MAE value of 0.2 means that the classifier predictions are, in average, 0.2 categories lower or higher than the target ones).

The approach based on HMMs [310] reports acceptable results but lower in general than those reported by the rest of methods. One possible reason is that the rest of the methods do not take into account the sequential character of wind speed and pressure values, while HMM does. Consequently, it is more difficult for HMMs to improve measures like C or MAE, than it is for the rest of more flexible methods.

From these tables, the Support Vector Machines methods seem to be the most competitive ones from all the different alternatives considered. When analysing the mean ranking and performance, the RED-SVM methodology obtains the better results for both measures. The second best methods are SVOREX and SVORIM for C, and SVORIM for MAE. Note that high accuracy values can be masking a lower ranking performance (i.e. a high MAE value), because the classifier can tend to assign rank values far from the real ones.

To determine the statistical significance of the rank differences observed for each method in the different datasets, a non-parametric Friedman test [247] has been car-

Table 7.3: Test Mean Absolute Error (MAE) results obtained by using the different methods evaluated.

	Wind farm						
Classifier	H	M	P	U	Z	$\overline{M}$	$\overline{R}_M$
SVM	**0.242**	*0.267*	0.365	0.382	*0.300*	0.311	2.90
LMT	0.250	0.293	0.459	0.383	0.373	0.352	6.20
C45	0.335	0.310	0.540	0.434	0.487	0.421	10.90
Ada10(C45)	0.314	0.318	0.492	*0.381*	0.420	0.385	8.60
Ada100(C45)	0.260	0.281	0.419	0.389	0.354	0.341	6.00
MLogistic	0.258	0.288	0.514	0.433	0.405	0.379	7.80
SLogistic	0.250	0.293	0.495	0.434	0.400	0.374	7.70
ASAOR(C45)	0.293	0.299	0.465	0.438	0.463	0.392	9.40
RED-SVM	**0.242**	**0.261**	0.364	0.382	**0.295**	**0.309**	**2.20**
SVOREX	*0.245*	0.268	**0.354**	**0.378**	0.317	*0.312*	2.70
SVORIM	0.248	*0.267*	*0.355*	**0.378**	0.314	*0.312*	*2.60*
GPOR	0.289	0.316	0.526	0.472	0.513	0.423	11.00
HMM	0.301	0.322	0.646	0.525	0.535	0.466	12.60

The best result is in bold face and the second best result in italics

ried out with the C and MAE rankings of the different methods (since a previous evaluation of the C and MAE values results in rejecting the normality and the equality of variances hypothesis). The test shows that the effect of the method used for classification is statistically significant at a significance level of $\alpha = 5\%$, as the confidence interval is $C_0 = (0, F_{0.05} = 1.96)$ and the F-distribution statistical values are $F^* = 12.37 \notin C_0$ for C and $F^* = 21.67 \notin C_0$ for MAE. As a result, the test concludes that all algorithms perform statistically differently in mean ranking.

The Bonferroni-Dunn test [240] is an approach to compare all classifiers to a given classifier (a control method), which is more sensitive than comparing all classifiers to each other. This test has been applied to both C and MAE rankings using RED-SVM as the control method. The test concludes that the differences in C and MAE values are significant:

- At a significance level of $\alpha = 5\%$, when RED-SVM is compared to C4.5, GPOR and HMM using the C measure (with C ranking differences of 8.30, 8.10, and 9.30, respectively) and to C4.5, ASA(C4.5), GPOR and HMM using the MAE measure (with MAE ranking differences of 8.90, 7.20, 8.80 and 10.40, respectively).
- Additionally, at a significance level of $\alpha = 10\%$, when RED-SVM is compared to ASA(C4.5) using the C measure (with C ranking difference of 6.80), and to Ada10(C4.5) using the MAE measure (with MAE ranking difference of 6.60).

It is important to outline that, although the rank differences are significant, the values obtained for the different measures (specially for accuracy, C) are very low (see Tables 7.2 and 7.3). Consequently, the study cannot clearly establish that the use of the ordering information improves the results obtained by the nominal classifiers. However, it should be mentioned that the number of ordinal methods which obtain better results is higher than in the nominal case, and that the differences for the MAE measure are generally higher.

7.4 CONCLUSIONS

This chapter introduced a new approach for daily mean wind speed series estimation, based on synoptic pressure measures. The problem has been stated as a classification task rather than the usual regression approach. Wind speed was discretized in four different ranges, which gather the main information needed by the experts when managing the wind farm. On the other hand, synoptic pressure measures in a grid have been considered as the input variables. The results of this preliminary study show that the best performing method is the SVM, with very high accuracy and low MAE values. Ordering information (more precisely, the RED and ASAOR algorithms) do not clearly outperform nominal methods (SVM and C4.5), given that very similar accuracies are obtained (although the differences in ranking over six datasets show to be significant).

Part IV

CONCLUSIONS

This final part of the thesis includes the main conclusions raised from all previous chapters and outlines some future research lines.

8

SUMMARY, CONCLUSIONS, AND FUTURE WORK

Summary. In this thesis, we have dealt with machine learning issues associated with the problem of ordinal classification. As stated in the introduction, the goal was to perform a review of the related research, to propose new learning methods for specific ordinal regression issues and to work in real world problems. In our opinion, these global goals have been achieved. To support this statement, this chapter finalizes the thesis with a summary of our contributions, together with some conclusions. We end the chapter with some future research directions.

Please note that more details about these conclusions are provided in corresponding chapters.

8.1 SUMMARY AND CONCLUSIONS

This thesis presents a research work in ordinal regression with respect to two issues of this field: class imbalance topic, which is shared with other classification problems, and class ordering exploitation to improve classifiers performance. In this section we summarize the thesis contributions grouped by topics.

8.1.1 *Literature review and methods taxonomy*

The thesis contribution begins with Chapter 3, which performs an exhaustive survey of the ordinal regression methods proposed in the literature. Up to the authors knowledge, there are not similar reviews in this field. That chapter first presents the problem setting, being clearly differentiated from other related topics. After this, a taxonomy of ordinal regression methods is proposed, dividing them into four main groups: naïve approaches, binary decompositions, threshold models and augmented binary classification.

We think the proposed taxonomy can assist future researchers to develop and propose new methods by helping them to categorize their contributions and to analize similar methods.

8.1.2 *Class imbalance*

Chapter 4 is devoted to the first major research topic in this thesis, which is the class imbalance. As introduced, in the recent years, and specially in the nominal classification area, this topic has motivated active research to produce classifiers which

are sensitive to all the classes, as well as to develop performance evaluation metrics that consider per-class classification throughput.

However, the issue of class imbalance has been mainly considered for the binary class case, due to a set of handicaps such as the difficulty of applying robust techniques, such as the ROC analysis to multi-class scenarios. Recently, the multi-class class imbalance issue has been tackled as a multi-objective optimization problem for the multi-class case. Nevertheless, the computational cost of this approach motivated further research to obtain less costly methods.

In Chapter 4 several alternatives are proposed to deal with multi-class imbalance in an efficient way, concluding that the one based on the $RMSE$ together with a probabilistic output was the most suitable (considering classification performance for all the classes and computational cost). Three keys were related to the success of the proposal. Firstly, the previous Pareto based approaches are reformulated as a weighed convex linear optimization problem. Secondly, the error functions are designed to produce continuous responses. Two continous alternatives were selected, one based on the cross-entropy error and the other based on the $RMSE$. The later being robust enough and computationally easier to calculate, it was selected for guiding an evolutionary algorithm. The first and the second key factors drastically reduced the cost of candidate solutions evaluation while helped to select more robust classifiers. The third fact for this efficient design is the use of the Evolutionary Extreme Learning Machine (E-ELM), which is an efficient algorithm for training feed-forward neural networks.

8.1.3 *Ordinal regression models and learning*

The second major research issue in this thesis is the exploration of new models and learning algorithms for ordinal regression. This is done in Chapter 5.

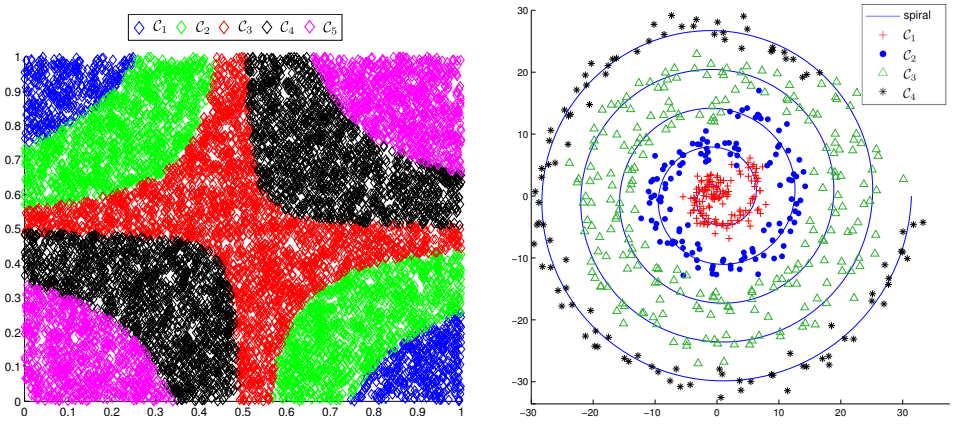
The first proposal is the Evolutionary Extreme Learning Machine for Ordinal Regression (E-ELMOR), which inherits lessons learned in Chapter 4 regarding model evaluation. In this way, the Weighed RMSE ($WRMSE$) is proposed to conduct candidate solutions selection in the evolutionary algorithm. The $WRMSE$ simultaneously reflects three criteria that are desirable for an ordinal classifier which also cares about the class imbalance issue: a) misclassification of non-adjacent classes should be more penalized as the difference between class labels grows; b) the posterior probability should be unimodal and monotonically decreasing for non-adjacent classes; c) in other error metrics such as MZE, only one network output (the one with maximum value) contributes to the error function, and it does not contribute with the output's value, whereas, in $RMSE$ based metrics each model output (posterior probabilities) contributes to the error function in such a way that the model's decision thresholds and posteriors will tend to be more discriminative. This is possible because of the use of a single multi-class model, given that this idea cannot be directly translated to binary decomposition schemes. The results reveal a remarkable performance improvement with respect to the reference methods.

The rest of the proposals aims at transforming the classification problem into a standard regression problem as a way of modelling the unknown latent variable which reflects the order arrangement between patterns of different classes.

In this line, the second proposal is the Numerical Variable Reconstruction (NVR). This alternative sticks to the strict definition of OR in which the order restriction is only applied to the labels, i.e. the output space. The method properly sets up several probability distributions to sample continuous values that are used to represent each pattern in the latent space. Depending of the class, the values are sampled from a different probability distribution, so that latent values for patterns of the same class rely on the same interval. Then, a regressor model is trained to predict this generated variable. Though NVR performs rightly for some datasets, it is not robust when extending the experiments to more data sets or when considering more performance metrics.

We considered that the idea of producing a continuous variable and training a regressor to predict that variable could still be valid, but this variable should be generated in a smarter way. Although the strict definition of OR does not consider input data order, the OR threshold models implicitly assume that the latent variable reflects somehow the total order of the patterns. Then it raised the idea of explicitly exploiting this order to construct this variable.

In order to do this, we want to capture the order of the dataset in the one-dimensional projection of the latent space, so relative positions of patterns in the input space should be translated into relative positions in the latent space. A first idea could be to use, for instance, the centroid of the class, so that the patterns can be placed in the latent space according to their distance to the centroid of the class. However, simply observing the two datasets represented in Figure 8.1, one may realize that this is not a robust idea, because in these datasets the centroid of all the classes is very similar. Therefore it raised the idea of using pairwise distances between neighbour classes for projecting the patterns into the one-dimensional space. In this way, we meet the order definition of OR in which the order is guaranteed between neighbour classes.



(a) Representation of *toy* dataset. (b) Representation of *spiral* dataset.

Figure 8.1: Representations of the *toy* dataset (by Herbrich et al. [8]) and the spiral dataset proposed in Chapter 5.

The idea expressed in the previous paragraph results in the proposed Pairwise Class Distance (PCD) projection, and then the associated classifier Pairwise Class Distances for Ordinal Classification (PCDOC). Experimentally, we concluded that the PCDOC method achieved suitable performance when compared to several state-of-

the-art methods. Additionally, we studied in detail the latent space organization of the projection based methods considered in that chapter. We suggest, that while the pressure for compactness within class latent projections can make training sample projections of the same class lie very close, for some datasets it can lead to poorer generalization overall performance. This issue is relaxed for the PCDOC algorithm which tends to produce softer projection models.

In conclusion, the results indicate that our two-phase approach to ordinal classification is a viable and simple-to-understand alternative to the state-of-art methods. In the case of PCDOC, the projection constructed in the first phase is consistently extracting useful information for ordinal classification. An example of the use of this information is done in the application of this technique to the sovereign credit rating problem in Chapter 6.

8.1.4 *Improvement of real applications under the umbrella of ordinal regression*

Finally, two real world application problems are presented.

The first application is presented in Chapter 6, and it is the rating of sovereign credit by using PCDOC, that is experimentally compared to other ordinal and nominal classifiers. The robustness of PCDOC, as well as that of other methods, is remarkable for several performance metrics.

In addition to the classification task, we use the projection of the regressor model in PCDOC for ranking visualization, which might be suitable to build a decision support system. In contrast to unsupervised visualization and projection techniques, this projection is validated by means of its suitability to correctly classify patterns.

The second application is the wind forecasting issue (Chapter 7). We propose to map the wind speed values to a set of ordinal labels in which each label is related to the power that can be generated associated to levels of wind speed. Extensive experiments are done comparing nominal and ordinal methods ability to predict the label which represents the level of wind.

The most important conclusion of these last two chapters is that addressing these two problems with ordinal regression techniques improved the generalization performance. This definitely justifies the current thesis and motivates further research in ordinal regression.

8.2 FUTURE WORK

There are several research lines that can extend the work presented in this thesis.

8.2.1 *Possible improvements for the proposed methods*

Regarding the methods proposed for ordinal regression, several issues can be further explored:

- In the case of E-ELMOR, future work could involve the design and experiments with new output codes and the associated error functions. For instance, new output codes for ANNs can be explored in order to consider the distance between class labels.

- Regarding the PCD projection, at the end of Chapter 5, there was a discussion point about the possible undesirable influence of outliers in the PCD projection. As suggested in the same chapter, a direct alternative could be to use k-NN like scheme for calculating the minimum distance of a pattern to patterns of the neighbour classes. In doing so, instead of taking the minimum distance to a point, the average distance to the k closest points of classes $q \pm 1$ could be used. This will represent a generalization of the current scheme that in the current form calculates distances with $k = 1$. Nevertheless, the inclusion of k would imply the addition of a new free parameter to the training process.

8.2.2 *The issue of data ordering evaluation*

Even though a problem nature suggests that there exist an order relationship between classes, several experiments in this thesis revealed that some of the nominal classifiers obtained better performance results in some (apparently) ordinal datasets than the ordinal classifiers.

As a base point for future research, in this section we perform some preliminary experiments in order to artificially disturb the class ordering. The purpose is to check whether the ordinal classifiers performance is affected if the order restriction is changed. For these experiments we simply relabel the datasets in order to modify the initial order arrangement of the classes. The relabelling consist on permutations of class labels. For each data set experiments are carried out with the original labels (*Original*), a random permutation of the labels (*Shuffle*), and an inverse ordering of the labels (*Inverse*). Table 8.1 shows the three label combinations used in the experiments depending on the number of classes of the problem and Table 8.2 presents the different datasets selected for this experiment.

Table 8.1: Original labelling and relabelling options ('random' labels (*Shuffle*) and inverse labels order (*Inverse*)). Note the second option is not a pure random execution since the order is altered with the restriction of avoiding partial orders between classes, and the 'random' label combination is the same for all the experiments to allow proper comparisons.

Number of classes	Original	Shuffle	Inverse
2	[1,2]	[2,1]	[2,1]
3	[1,2,3]	[1,3,2]	[3,2,1]
4	[1,2,3,4]	[1,4,2,3]	[4,3,2,1]
5	[1,2,3,4,5]	[3,1,5,2,4]	[5,4,3,2,1]
6	[1,2,3,4,5,6]	[1,5,2,4,6,3]	[6,5,4,3,2,1]
7	[1,2,3,4,5,6,7]	[3,1,7,5,2,6,4]	[7,6,5,4,3,2,1]
8	[1,2,3,4,5,6,7,8]	[3,8,1,7,5,2,6,4]	[8,7,6,5,4,3,2,1]
9	[1,2,3,4,5,6,7,8,9]	[3,8,1,7,5,9,2,6,4]	[9,8,7,6,5,4,3,2,1]
10	[1,2,3,4,5,6,7,8,9,10]	[3,10,8,1,7,5,9,2,6,4]	[10,9,8,7,6,5,4,3,2,1]

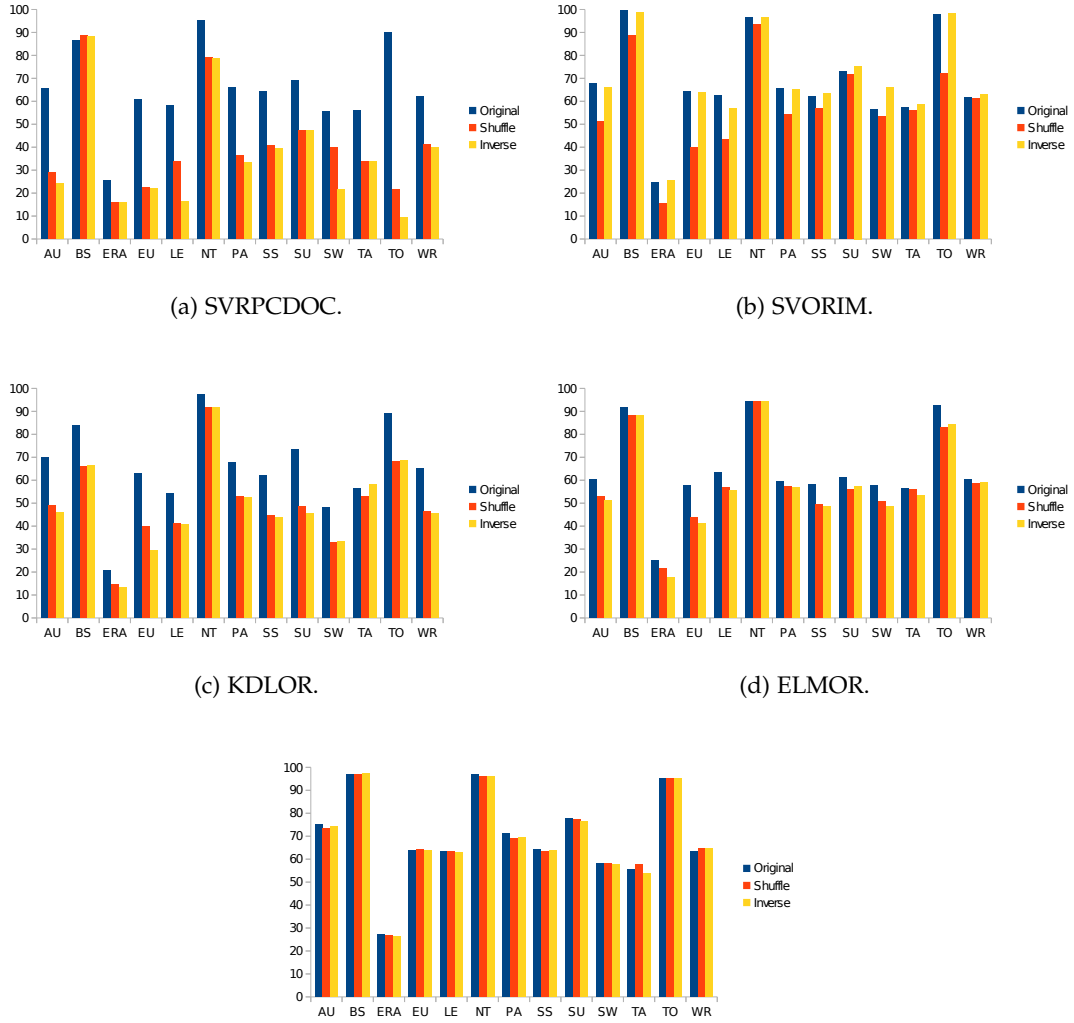
Table 8.2: Characteristics of the benchmark datasets considered for the relabelling experiment

Dataset	#Pat.	#Attr.	#Classes	Class distribution
automobile (AU)	205	71	6	(3, 22, 67, 54, 32, 27)
balance-scale (BS)	625	4	3	(288, 49, 288)
ERA (ER)	1000	4	9	(92, 142, 181, 172, 158, 118, 88, 31, 18)
eucalyptus (EU)	736	91	5	(180, 107, 130, 214, 105)
LEV (LE)	1000	4	5	(93, 280, 403, 197, 27)
newthyroid (NT)	215	5	3	(30, 150, 35)
pasture (PA)	36	25	3	(12, 12, 12)
squash-stored (SS)	52	51	3	(23, 21, 8)
squash-unstored (SU)	52	52	3	(24, 24, 4)
SWD (SW)	1000	10	4	(32, 352, 399, 217)
tae (TA)	151	54	3	(49, 50, 52)
toy (TO)	300	2	5	(35, 87, 79, 68, 31)
winequality-red (WR)	1599	11	6	(10, 53, 681, 638, 199, 18)

The experiments where carried out in a similar way as in the rest of the thesis, and the mean generalization *Acc* performance is used for comparison purposes. Note that ordinal regression metrics are not used here since we do not assume ordering between classes anymore. Also *Acc* is the metric used as hyper-parameters selection criteria.

Figure 8.2 shows the performance of several classifiers in the same datasets with different labels ordering. This figure reveals that some of the ordinal regression methods have a remarkable performance drop. Specifically, the thresholds methods (PCDOC, SVORIM and KDLOR) are negatively affected by the random relabelling. In the case of PCDOC and KDLOR, the performance is also degraded for the inverse labelling, while SVORIM is not affected by this relabelling. ELMOR is also affected for the targets relabelling, however, its performance decay is smaller. As expected, the SVC is not affected by relabelling[1]. From this study, the questions to be answered could be: a) why is the performance decay higher for some datasets?, and b) why are some OR methods more robust to this relabelling process?. The answer to a) is obviously related to the fact that ordinal structure can be found in the label space but not in the input space. Consequently, designing a method to evaluate the degree of ordinality of a dataset in the input space is a possible future research line. The

1 Note that though SVC is a deterministic method there are small variations in the performance. The reason is that the hyper-parameters optimization used a grid search with an internal 5-fold cross-validation procedure. This 5-fold performs different partitions of the training set, producing training and validation sets, and the partitions depends on a random seed. Specially for small datasets, the selection of some patterns for the training or validation sets can influence the final hyper-parameters' values selection, and consequently can affect generalization performance.



(a) SVRPCDOC.

(b) SVORIM.

(c) KDLOR.

(d) ELMOR.

(e) SVC.

Figure 8.2: Preliminary experiments showing *Acc* performance for different methods with original labels (*Original*), 'random' labels (*Shuffle*) and inverse labels order (*Inverse*).

answer to b) should be found in the way each algorithm constructs and optimizes the different classifiers.

In conclusion, and considering the preliminary experiments presented in this section, the motivation of performing input space analysis in order to evaluate the ordinality degree of the problem relies here. Then, the future work road map should definitely include methods for assessing the suitability of addressing the problem as an ordinal regression task or as a nominal one, depending on previous evaluations of the data set.

APPENDIX

A

EXPERIMENTAL RESULTS TABLES FOR CHAPTER 4

A.1 STATISTICAL RESULTS TABLES FOR C, MS AND T

Table A.1: Statistical results for C, MS and T

Dataset	Algorithm	C	MS	T
anneal	EELMCS(R)	96.46 ± 01.75	80.43 ± 11.13	1.71E+002 ± 1.84E+000
	EELMCS(E)	95.48 ± 01.93	76.76 ± 11.14	1.79E+002 ± 3.99E+000
	EELMCS(CS)	*98.07 ± 01.71*	**89.01 ± 11.49**	1.62E+002 ± 2.59E+000
	EELM(C)	**99.11 ± 00.90**	59.95 ± 47.29	1.67E+002 ± 7.36E+000
	EELM(MS)	96.07 ± 02.65	*83.61 ± 13.90*	1.61E+002 ± 2.47E+000
	OPELM	95.87 ± 01.21	51.67 ± 06.48	5.97E+000 ± 8.86E-001
	ELM	96.74 ± 01.01	55.60 ± 12.83	*2.26E-001 ± 9.96E-002*
	PDE(C)	90.24 ± 03.23	34.40 ± 27.22	3.02E+003 ± 3.92E+002
	PDE(MS)	86.22 ± 06.42	53.76 ± 15.37	3.02E+003 ± 3.92E+002
	HPDE(C)	92.22 ± 02.56	41.40 ± 27.91	3.01E+003 ± 4.77E+002
	HPDE(MS)	87.08 ± 06.81	59.14 ± 14.01	3.01E+003 ± 4.77E+002
	Rprop	95.82 ± 00.03	19.47 ± 00.37	2.24E+000 ± 8.14E-001
	SVC	97.78 ± 00.00	50.00 ± 00.00	**4.84E-002 ± 0.00E+000**
balance	EELMCS(R)	91.65 ± 00.94	**87.67 ± 06.83**	6.14E+001 ± 2.41E+000
	EELMCS(E)	91.86 ± 00.79	*87.42 ± 06.78*	6.38E+001 ± 2.08E+000
	EELMCS(CS)	90.92 ± 01.47	83.32 ± 10.85	6.07E+001 ± 1.29E+000
	EELM(C)	91.32 ± 01.70	36.33 ± 26.46	6.10E+001 ± 1.67E+000
	EELM(MS)	90.49 ± 02.00	81.86 ± 19.57	5.91E+001 ± 3.10E+000
	OPELM	91.97 ± 01.61	16.33 ± 18.29	6.89E+000 ± 1.17E+000
	ELM	88.55 ± 01.39	06.67 ± 06.06	*3.54E-001 ± 1.07E-001*
	PDE(C)	90.36 ± 01.30	23.33 ± 16.68	1.03E+002 ± 1.03E+001
	PDE(MS)	91.05 ± 01.15	85.15 ± 07.68	1.03E+002 ± 1.03E+001
	HPDE(C)	91.24 ± 01.23	29.00 ± 11.85	1.20E+002 ± 1.34E+001
	HPDE(MS)	91.22 ± 01.49	84.62 ± 06.82	1.20E+002 ± 1.34E+001
	Rprop	*92.56 ± 00.04*	17.33 ± 00.27	8.41E-001 ± 2.89E-001
	SVC	**93.59 ± 00.00**	80.00 ± 00.00	**1.19E-002 ± 0.00E+000**
breastw	EELMCS(R)	96.30 ± 00.93	**94.46 ± 02.07**	1.77E+001 ± 3.43E-001
	EELMCS(E)	96.02 ± 00.79	93.24 ± 02.11	1.88E+001 ± 4.19E-001
	EELMCS(CS)	96.23 ± 00.86	93.54 ± 02.23	1.79E+001 ± 3.90E-001
	EELM(C)	*96.38 ± 00.77*	*94.42 ± 01.79*	1.72E+001 ± 4.41E-001
	EELM(MS)	95.70 ± 00.92	91.69 ± 02.52	1.73E+001 ± 3.60E-001
	OPELM	95.92 ± 00.88	93.76 ± 01.98	1.40E+000 ± 2.97E-001
	ELM	95.81 ± 00.66	92.06 ± 01.89	*1.96E-002 ± 2.53E-002*

Continued on Next Page...

Dataset	Algorithm	*C*	*MS*	*T*
	PDE(C)	95.14 ± 00.99	90.22 ± 02.69	2.47E+001 ± 3.56E+000
	PDE(MS)	95.18 ± 00.93	90.33 ± 02.85	2.47E+001 ± 3.56E+000
	HPDE(C)	95.03 ± 00.86	89.56 ± 02.19	2.92E+001 ± 3.77E+000
	HPDE(MS)	94.93 ± 00.85	89.44 ± 02.16	2.92E+001 ± 3.77E+000
	Rprop	96.13 ± 00.01	93.43 ± 00.02	4.78E-001 ± 1.46E-001
	SVC	**96.57 ± 00.00**	91.67 ± 00.00	**7.26E-003 ± 0.00E+000**
card	EELMCS(R)	88.07 ± 01.64	**86.09 ± 01.59**	4.95E+001 ± 3.95E-001
	EELMCS(E)	*88.13 ± 01.55*	*85.95 ± 03.12*	5.22E+001 ± 3.43E-001
	EELMCS(CS)	87.15 ± 01.94	85.59 ± 02.25	4.85E+001 ± 1.59E+000
	EELM(C)	86.92 ± 01.50	84.82 ± 02.50	4.74E+001 ± 8.91E-001
	EELM(MS)	87.23 ± 01.62	85.02 ± 02.28	4.86E+001 ± 1.39E+000
	OPELM	84.84 ± 01.84	82.93 ± 02.72	6.21E+000 ± 7.78E-001
	ELM	85.53 ± 01.93	84.33 ± 02.03	*5.02E-002 ± 5.34E-002*
	PDE(C)	85.39 ± 02.32	83.01 ± 03.08	2.98E+001 ± 7.36E+000
	PDE(MS)	85.74 ± 02.06	84.08 ± 02.44	2.98E+001 ± 7.36E+000
	HPDE(C)	86.55 ± 01.14	84.85 ± 02.25	5.37E+001 ± 1.10E+001
	HPDE(MS)	86.51 ± 01.32	85.30 ± 01.83	5.37E+001 ± 1.10E+001
	Rprop	86.19 ± 00.08	82.85 ± 00.16	6.06E-001 ± 1.49E-001
	SVC	**88.44 ± 00.00**	85.42 ± 00.00	**3.31E-002 ± 0.00E+000**
gene	EELMCS(R)	*84.73 ± 01.15*	78.96 ± 02.14	9.79E+002 ± 8.23E+001
	EELMCS(E)	84.67 ± 01.16	78.20 ± 03.59	9.99E+002 ± 8.63E+001
	EELMCS(CS)	83.46 ± 01.30	78.84 ± 03.80	9.42E+002 ± 8.57E+001
	EELM(C)	84.50 ± 01.45	78.52 ± 03.34	9.61E+002 ± 8.95E+001
	EELM(MS)	83.69 ± 01.65	*79.57 ± 02.60*	9.61E+002 ± 9.05E+001
	OPELM	77.99 ± 01.41	62.86 ± 04.43	8.56E+000 ± 1.06E-001
	ELM	80.50 ± 01.33	69.65 ± 03.23	**2.37E-001 ± 2.09E-002**
	PDE(C)	70.37 ± 04.18	56.74 ± 09.93	1.71E+004 ± 4.38E+003
	PDE(MS)	69.99 ± 04.34	65.25 ± 05.83	1.71E+004 ± 4.38E+003
	HPDE(C)	82.32 ± 02.83	75.15 ± 05.43	1.37E+007 ± 2.70E+006
	HPDE(MS)	82.06 ± 02.76	74.89 ± 05.47	1.37E+007 ± 2.70E+006
	Rprop	84.39 ± 00.07	73.52 ± 00.23	2.17E+000 ± 5.43E-001
	SVC	**90.92 ± 00.00**	**89.32 ± 00.00**	*1.33E+000 ± 0.00E+000*
glass	EELMCS(R)	62.83 ± 07.82	**19.17 ± 16.54**	7.31E+001 ± 9.53E-001
	EELMCS(E)	69.62 ± 04.16	05.00 ± 12.11	7.52E+001 ± 1.05E+000
	EELMCS(CS)	68.55 ± 04.41	*18.61 ± 17.46*	7.87E+001 ± 1.37E+000
	EELM(C)	69.69 ± 05.22	07.78 ± 13.83	7.81E+001 ± 1.64E+000
	EELM(MS)	66.42 ± 06.08	15.47 ± 17.16	7.74E+001 ± 1.76E+000
	OPELM	**71.70 ± 03.43**	02.50 ± 07.63	2.88E+000 ± 4.94E-001
	ELM	*70.44 ± 04.86*	00.00 ± 00.00	**4.38E-003 ± 1.42E-002**
	PDE(C)	61.89 ± 08.53	01.67 ± 06.34	3.26E+002 ± 5.04E+001
	PDE(MS)	57.99 ± 09.11	09.46 ± 15.06	3.26E+002 ± 5.04E+001
	HPDE(C)	69.18 ± 05.23	02.50 ± 07.63	3.61E+002 ± 7.80E+001
	HPDE(MS)	64.53 ± 07.41	18.43 ± 17.96	3.61E+002 ± 7.80E+001
	Rprop	59.69 ± 00.11	00.00 ± 00.00	5.52E-001 ± 1.80E-001
	SVC	64.15 ± 00.00	00.00 ± 00.00	*8.35E-003 ± 0.00E+000*
hepatitis	EELMCS(R)	74.27 ± 04.23	42.92 ± 13.80	1.71E+000 ± 8.55E-002
	EELMCS(E)	76.24 ± 04.26	38.75 ± 14.44	1.70E+000 ± 9.52E-003

Continued on Next Page...

Dataset	Algorithm	*C*	*MS*	*T*
	EELMCS(CS)	*77.18 ± 04.43*	*45.90 ± 12.95*	1.67E+000 ± 1.06E-002
	EELM(C)	76.41 ± 04.64	35.00 ± 15.54	1.67E+000 ± 8.34E-003
	EELM(MS)	74.87 ± 04.92	41.25 ± 13.99	1.67E+000 ± 9.07E-003
	OPELM	76.32 ± 03.79	30.00 ± 14.53	2.27E-001 ± 6.25E-003
	ELM	76.75 ± 03.92	32.50 ± 11.65	**1.53E-003 ± 3.65E-005**
	PDE(C)	74.70 ± 04.24	33.33 ± 13.67	1.14E+001 ± 1.78E+000
	PDE(MS)	74.70 ± 04.35	35.42 ± 12.32	1.14E+001 ± 1.78E+000
	HPDE(C)	75.38 ± 03.34	31.67 ± 12.17	1.40E+001 ± 3.38E+000
	HPDE(MS)	75.21 ± 04.11	33.33 ± 12.43	1.40E+001 ± 3.38E+000
	Rprop	76.07 ± 00.04	22.08 ± 00.20	2.59E-001 ± 8.00E-002
	SVC	**79.49 ± 00.00**	**50.00 ± 00.00**	*1.58E-003 ± 0.00E+000*
iris	EELMCS(R)	96.00 ± 01.48	89.16 ± 03.29	1.76E+000 ± 2.17E-002
	EELMCS(E)	95.70 ± 01.64	88.65 ± 03.08	1.82E+000 ± 1.78E-002
	EELMCS(CS)	94.96 ± 01.93	87.56 ± 03.81	1.80E+000 ± 8.17E-003
	EELM(C)	95.26 ± 02.16	89.09 ± 04.04	1.81E+000 ± 1.10E-002
	EELM(MS)	94.74 ± 01.89	87.32 ± 03.63	1.80E+000 ± 1.07E-002
	OPELM	92.89 ± 03.16	86.98 ± 07.09	8.08E-001 ± 2.09E-001
	ELM	95.70 ± 02.18	90.54 ± 05.07	*1.29E-002 ± 1.55E-002*
	PDE(C)	96.58 ± 01.82	90.26 ± 04.01	7.98E+000 ± 9.75E-001
	PDE(MS)	95.81 ± 01.43	87.69 ± 04.33	7.98E+000 ± 9.75E-001
	HPDE(C)	*97.09 ± 00.89*	*91.28 ± 02.66*	9.89E+000 ± 2.43E+000
	HPDE(MS)	95.73 ± 01.23	87.18 ± 03.69	9.89E+000 ± 2.43E+000
	Rprop	90.00 ± 00.13	71.76 ± 00.36	4.20E-001 ± 1.21E-001
	SVC	**97.78 ± 00.00**	**93.33 ± 00.00**	**6.18E-004 ± 0.00E+000**
lymph	EELMCS(R)	79.82 ± 03.74	04.83 ± 18.41	2.95E+001 ± 2.00E-001
	EELMCS(E)	78.02 ± 05.35	02.33 ± 12.78	3.10E+001 ± 2.06E-001
	EELMCS(CS)	77.75 ± 05.10	02.67 ± 14.61	3.11E+001 ± 3.93E-001
	EELM(C)	79.01 ± 05.10	00.00 ± 00.00	3.13E+001 ± 3.64E-001
	EELM(MS)	70.45 ± 06.46	06.28 ± 19.29	3.12E+001 ± 2.55E-001
	OPELM	*82.43 ± 04.36*	00.00 ± 00.00	1.79E-001 ± 4.19E-003
	ELM	79.28 ± 04.83	07.00 ± 21.36	*4.85E-003 ± 1.22E-004*
	PDE(C)	79.46 ± 04.63	02.33 ± 12.78	9.75E+001 ± 2.67E+001
	PDE(MS)	79.28 ± 04.78	02.33 ± 12.78	9.75E+001 ± 2.67E+001
	HPDE(C)	80.99 ± 05.74	**14.83 ± 30.44**	1.21E+002 ± 2.57E+001
	HPDE(MS)	80.81 ± 05.74	**14.83 ± 30.44**	1.21E+002 ± 2.57E+001
	Rprop	80.99 ± 00.10	00.00 ± 00.00	3.21E-001 ± 1.29E-001
	SVC	**83.78 ± 00.00**	00.00 ± 00.00	**2.84E-003 ± 0.00E+000**
zoo	EELMCS(R)	91.73 ± 04.06	**08.89 ± 18.69**	3.61E+001 ± 3.84E-001
	EELMCS(E)	*92.93 ± 04.54*	**08.89 ± 27.59**	3.76E+001 ± 2.98E-001
	EELMCS(CS)	90.40 ± 04.15	00.00 ± 00.00	4.22E+001 ± 3.24E-001
	EELM(C)	89.20 ± 03.95	02.22 ± 12.17	4.22E+001 ± 3.83E-001
	EELM(MS)	88.93 ± 04.42	03.33 ± 18.26	4.23E+001 ± 4.78E-001
	OPELM	87.20 ± 05.79	00.00 ± 00.00	4.60E-002 ± 7.87E-004
	ELM	91.60 ± 04.74	05.00 ± 20.13	*1.68E-003 ± 1.30E-004*
	PDE(C)	85.07 ± 06.12	02.78 ± 10.80	6.15E+001 ± 1.68E+001
	PDE(MS)	84.67 ± 06.31	02.78 ± 10.80	6.15E+001 ± 1.68E+001
	HPDE(C)	90.67 ± 05.49	**08.89 ± 27.59**	7.98E+001 ± 3.89E+001

Continued on Next Page...

Dataset	Algorithm	*C*	*MS*	*T*
	HPDE(MS)	90.80 ± 05.37	**08.89 ± 27.59**	7.98E+001 ± 3.89E+001
	Rprop	86.93 ± 00.12	00.00 ± 00.00	4.98E-001 ± 2.04E-001
	SVC	**96.00 ± 00.00**	00.00 ± 00.00	**1.50E-003 ± 0.00E+000**

B

SYNTHETIC DATA GENERATION FOR ORDINAL REGRESSION

In this appendix we describe the developed synthetic data generator for creating ordinal regression datasets. We focus on the general case dealing with challenging data complexity topics that are interesting in general, and more specifically, in the context of ordinal classification. Those topics are: class ordering in the input space, data dimensionality, class overlapping and data multimodality.

The program is written in Matlab language and it allows the generation of datasets with multidimensional isotropic Gaussian – also known as white Gaussian – distributions presenting the following parameters: number of classes, number of patterns per class, number of input dimensions, variance of the Gaussians and number of modes for each class. The framework allows data visualization which can be exported in PDF and SVG formats (via Matlab toolboxes), and datasets files can be exported to Weka and Matlab formats. The source code of the synthetic data generator is available on a public website[1] and it is released under the GNU General Public License version 3 (GPLv3) [253].

Associated publications. Some portions of this appendix appeared in the following publication [242].

B.1 ISOTROPIC GAUSSIAN SYNTHETIC DATA GENERATION

In order to simplify the requirements of the data generation, the patterns will be generated by random sampling from isotropic Gaussian distributions (i.e. distribution variance is the same through all the dimensions). Being, $\mathcal{N}(\mu, \sigma^2)$ a one-dimension Gaussian distribution, where μ is the mean and σ^2 is the variance. For higher dimensionality, the multivariate Gaussian distribution is defined as $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{\mu}$ is a n-dimensional mean vector, and $\boldsymbol{\Sigma}_{n\times n}$ is the covariance matrix. In the case of the multivariate isotropic Gaussian distribution, the distribution can be expressed as:

$$\mathbf{x} = \mathcal{N}(\boldsymbol{\mu}, \sigma^2 \mathbf{I}_{n\times n}), \tag{B.1}$$

where sample $\mathbf{x} \in \mathbb{X} \subseteq \mathbb{R}^n$ and $\mathbf{I}_{n\times n}$ is the identity matrix. Note this formulation reflects that the variance σ^2 is the same across all dimensions and that all the dimensions' variance are independent.

From now on, we will work with the multidimensional isotropic Gaussian distributions (Eq. (B.1)). With these premises, a hyper-sphere or n-sphere with center in $\mathbf{x}$ and radio r can be defined as follows:

1 http://www.uco.es/grupos/ayrna/iwann2013-syntheticdatagenerator

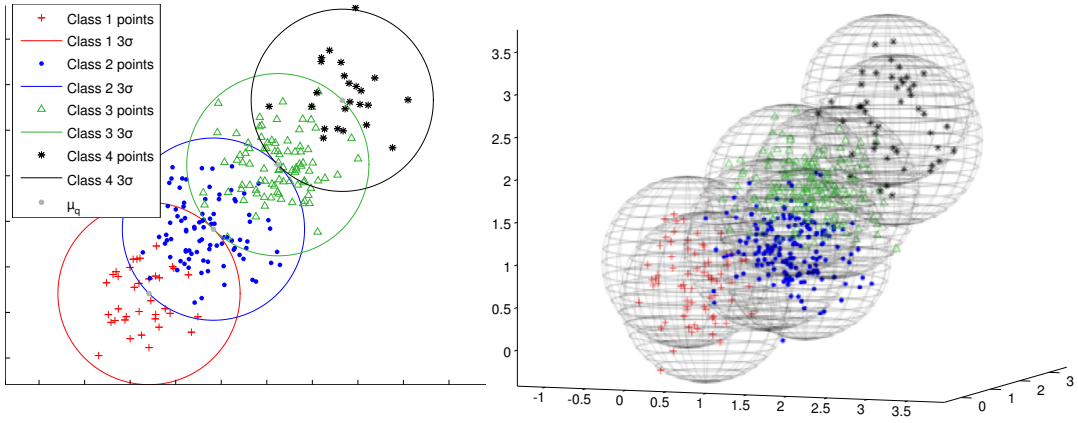


Figure B.1: Example of two synthetic datasets. The left dataset has $n = 2$, $\sigma^2 = 0.33$ and $m = 1$. The right dataset is generated with $n = 3$, $\sigma^2 = 0.33$ and $m = 2$.

$$S^n = \left\{ \mathbf{x} \in \mathbb{R}^{n+1} : \|\mathbf{x}\| = r \right\}, \tag{B.2}$$

in our case, $\mathbf{x} = \boldsymbol{\mu}$. Considering $r = 3\sigma$, nearly all (99.73%) of the population lies inside the n-sphere.

Then, we want to place the set of n-spheres with a separation between them of α in a Euclidean space, this is, the distance between one center $\boldsymbol{\mu}$ and another center $\boldsymbol{\mu}' = \boldsymbol{\mu} + \Delta_{\boldsymbol{\mu}}$ will be α:

$$\mathrm{d}(\boldsymbol{\mu}, \boldsymbol{\mu}') = \sqrt{\sum_{i=1}^{n} (\mu_i - \mu_i')^2} = \sqrt{\sum_{i=1}^{n} (\mu_i - \mu_i + \Delta_{\boldsymbol{\mu}})^2} = \alpha, \tag{B.3}$$

$$\Delta_{\boldsymbol{\mu}} = \pm \alpha / \sqrt{n}. \tag{B.4}$$

Then, using an increment of $\Delta_{\boldsymbol{\mu}}$ guarantees a separation of α in the Euclidean space independently of the dimensionality. With this separation between the centers of the n-spheres the percentage of overlapped surface ($n = 2$) or volume ($n \geq 3$) respecting the n-sphere will be constant. Then, the overlap can be expressed in terms of σ. Note that working with anisotropic Gaussian would imply defining multiple hyper-ellipses, thus dealing with more complex calculations to effectively control class overlapping in multiple dimensions.

Regarding multimodality of the data, we proceed in the following way: the previously defined Gaussian distribution is considered as the *main* distribution of each class, and additional distributions are added to each class in order force multimodality. For each class q, the additional Gaussian distributions are centered in a random location within the surface of the n-sphere with center $\boldsymbol{\mu}_q$ and radius Δ_r. In order to sample points only on the n-sphere surface, the norm of the samples is used as the n-space position (see Eq. (B.2)), then, the center of each additional mode is obtained as:

$$\boldsymbol{\mu}_q^i = \Delta_r \times \|\mathbf{x}\| \, , \mathbf{x} \in C_q, \tag{B.5}$$

where $\boldsymbol{\mu}_q^i$ is the i-th mode of the q-th class, $\mathbf{x}$ is sampled from Eq. (B.1) and $\Delta_r = \pm\lambda/\sqrt{n}$, being $\lambda < \alpha$ the desired separation between each mode of the class. In this way, each class can be composed of different modes, however the overlap of the additional distributions can not be controlled. We denote the number of modes per class with m. Figure B.1 shows an example of two generated datasets and theoretical n-spheres.

BIBLIOGRAPHY

[1] Tjen-Sien Lim, Wei-Yin Loh, and Yu-Shan Shih. A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms. *Machine Learning*, 40:203–228, September 2000. ISSN 0885-6125. (Cited on pages ix, xi, 7, and 28.)

[2] Dennis F Kibler, David W Aha, and Marc K Albert. Instance-based prediction of real-valued attributes. *Computational Intelligence*, 5:51, 1989. (Cited on pages ix, xi, 7, and 28.)

[3] Dave Barker. Pasture Production dataset, 1995. URL http://mldata.org/repository/data/viewslug/agridatasets-pasture/. Obtained on October 2011. (Cited on pages ix, xi, 7, and 28.)

[4] J.S. Cardoso, J.F. Pinto da Costa, and M.J. Cardoso. Modelling ordinal relations with SVMs: an application to objective aesthetic evaluation of breast cancer conservative treatment. *Neural Networks*, 18(5-6):808–817, 2005. (Cited on pages ix, xi, 7, 28, 56, and 74.)

[5] P.A. Gutiérrez, S. Salcedo-Sanz, C. Hervás-Martínez, L. Carro-Calvo, J. Sánchez-Monedero, and L. Prieto. Ordinal and nominal classification of wind speed from synoptic pressure patterns. *Engineering Applications of Artificial Intelligence*, 26(3):1008–1015, 2013. (Cited on pages ix, xi, 56, and 161.)

[6] Kyoung-jae Kim and Hyunchul Ahn. A corporate credit rating model using multi-class support vector machines with an ordinal pairwise partitioning approach. *Computers & Operations Research*, 39(8):1800–1811, 2012. (Cited on pages ix, xi, 7, 28, 56, 66, and 150.)

[7] Robert Arens. Learning SVM Ranking Functions from User Feedback Using Document Metadata and Active Learning in the Biomedical Domain. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 363–383. Springer-Verlag, 2010. (Cited on pages xi, 7, and 28.)

[8] R. Herbrich, T. Graepel, and K. Obermayer. Support vector learning for ordinal regression. In *Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference on (Conf. Publ. No. 470)*, volume 1, pages 97–102, 1999. (Cited on pages xxiii, xxv, 17, 70, and 177.)

[9] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 1st ed. 2006. corr. 2nd printing edition, August 2007. ISBN 0387310738. (Cited on pages xxiii, 40, 42, 43, 49, 50, 51, 52, 63, 67, 72, 89, and 91.)

[10] Andrew Ng. Stanford's machine learning course, 2011. URL http://www.holehouse.org/mlclass/index.html. [Online; accessed 15-March-2013]. (Cited on pages 3, 5, 25, and 26.)

[11] Wikipedia. Machine learning — wikipedia, the free encyclopedia, 2013. URL http://en.wikipedia.org/w/index.php?title=Machine_learning&oldid=544340996. [Online; accessed 15-March-2013]. (Cited on pages 3 and 25.)

[12] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997. ISBN ISBN 0-07-042807-7. (Cited on pages 3 and 25.)

[13] Gavin Brown and Ke Chen. Machine learning. lecture: Motivation – what is machine learning?, 2011. URL http://www.cs.manchester.ac.uk/ugt/COMP24111/. [Online; accessed 15-March-2013]. (Cited on pages 3, 4, 25, and 26.)

[14] Włodzisław Duch. What is computational intelligence and where is it going? In Włodzisław Duch and Jacek Mańdziuk, editors, *Challenges for Computational Intelligence*, volume 63 of *Studies in Computational Intelligence*, pages 1–13. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-71983-0. (Cited on pages 4 and 25.)

[15] Wikipedia. Artificial neural network — wikipedia, the free encyclopedia, 2013. URL http://en.wikipedia.org/w/index.php?title=Artificial_neural_network&oldid=547319984. [Online; accessed 31-March-2013]. (Cited on pages 4, 25, and 38.)

[16] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, USA, 1 edition, January 1996. ISBN 0198538642. (Cited on pages 4, 25, 38, 88, and 91.)

[17] Jens C. Hühn and Eyke Hüllermeier. Is an ordinal class structure useful in classifier learning? *Int. J. of Data Mining, Modelling and Management*, 1(1):45–67, 2008. (Cited on pages 6, 7, 10, 12, 27, 28, 31, 33, 57, and 106.)

[18] Joaquim F. Pinto da Costa, Hugo Alonso, and Jaime S. Cardoso. The unimodal model for the classification of ordinal data. *Neural Networks*, 21:78–91, January 2008. ISSN 0893-6080. (Cited on pages 6, 8, 27, 28, 56, 60, 68, 75, and 108.)

[19] M. Cruz-Ramírez, C. Hervás-Martínez, J. Sánchez-Monedero, and P.A. Gutiérrez. A Preliminary Study of Ordinal Metrics to Guide a Multi-Objective Evolutionary Algorithm. In *Proceedings of the 11th International Conference on Intelligent Systems Design and Applications (ISDA 2011)*, pages 1176–1181, Cordoba, Spain, 2011. (Cited on pages 10, 31, 55, and 56.)

[20] J. Sánchez-Monedero M. Cruz-Ramírez, C. Hervás-Martínez and P.A. Gutiérrez. Metrics to guide a multi-objective evolutionary algorithm for ordinal classification. *Neurocomputing*, Accepted, 2013. (Cited on pages 8 and 28.)

[21] Juan C. Fernández-Caballero, Francisco J. Martínez-Estudillo, César Hervás-Martínez, and Pedro A. Gutiérrez. Sensitivity versus accuracy in multiclass problems using memetic pareto evolutionary neural networks. *IEEE Transactions on Neural Networks*, 21(5):750 –770, may 2010. ISSN 1045-9227. (Cited on pages 8, 12, 28, 33, 79, 80, 85, 87, 88, and 90.)

[22] P.A. Gutiérrez, C. Hervás-Martínez, F. Martínez-Estudillo, and M. Carbonero-Ruz. A two-stage evolutionary algorithm based on sensitivity and accuracy

for multi-class problems. *Information Sciences*, 197:20–37, 2012. ISSN 0020-0255. (Cited on pages 8, 12, 28, and 33.)

[23] S. Baccianella, A. Esuli, and F. Sebastiani. Evaluation measures for ordinal regression. In *Proceedings of the Ninth International Conference on Intelligent Systems Design and Applications (ISDA'09)*, pages 283–287, San Mateo, CA, 2009. IEEE Computer Society. (Cited on pages 8, 29, and 60.)

[24] Vikas C. Raykar, Ramani Duraiswami, and Balaji Krishnapuram. A fast algorithm for learning a ranking function from large-scale data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1158–1170, 2008. ISSN 0162-8828. (Cited on pages 8, 29, and 59.)

[25] Stefan Kramer, Gerhard Widmer, Bernhard Pfahringer, and Michael de Groeve. Prediction of ordinal classes using regression trees. In Zbigniew Ras and Setsuo Ohsuga, editors, *Foundations of Intelligent Systems*, volume 1932 of *Lecture Notes in Computer Science*, pages 665–674. Springer Berlin / Heidelberg, 2010. (Cited on pages 8, 12, 29, 33, and 63.)

[26] Sotiris B. Kotsiantis and Panayiotis E. Pintelas. A cost sensitive technique for ordinal classification problems. In George Vouros and Themistoklis Panayiotopoulos, editors, *Methods and applications of artificial intelligence (Proceedings of the Third Hellenic Conference on Artificial Intelligence, SETN 2004)*, volume 3025 of *Lecture Notes in Artificial Intelligence*, pages 220–229, 2004. (Cited on pages 8, 29, and 64.)

[27] Eibe Frank and Mark Hall. A simple approach to ordinal classification. In *Proceedings of the 12th European Conference on Machine Learning*, EMCL '01, pages 145–156, London, UK, 2001. Springer-Verlag. (Cited on pages 8, 29, 30, 65, 76, 107, 110, 132, 154, and 167.)

[28] Willem Waegeman and Luc Boullart. An ensemble of weighted support vector machines for ordinal regression. *International Journal of Computer Systems Science and Engineering*, 3(1):47–51, 2009. (Cited on pages 8, 29, 30, and 66.)

[29] Jaime S. Cardoso and Joaquim F. Pinto da Costa. Learning to classify ordinal data: The data replication method. *Journal of Machine Learning Research*, 8:1393–1429, 2007. (Cited on pages 9, 30, 65, 68, 71, 74, and 75.)

[30] L. Li and H.-T. Lin. Ordinal regression by extended binary classification. In *Advances in Neural Information Processing Systems*, volume 2, pages 865–872, 2006. (Cited on pages 9, 30, 60, 73, 76, 118, 132, 154, and 167.)

[31] Hsuan-Tien Lin and Ling Li. Reduction from cost-sensitive ordinal ranking to weighted binary classification. *Neural Computation*, 24(5):1329–1367, 2012. (Cited on pages 9, 30, 58, 59, 64, 65, 71, 73, 74, 76, 118, and 132.)

[32] K. Crammer and Y. Singer. Online ranking by projecting. *Neural Computation*, 17(1):145–175, 2005. (Cited on pages 9, 30, 72, and 74.)

[33] Wei Chu and S. Sathiya Keerthi. Support Vector Ordinal Regression. *Neural Computation*, 19(3):792–815, 2007. (Cited on pages 9, 30, 56, 60, 68, 71, 74, 76, 118, 127, 132, 154, and 167.)

[34] Shereen Fouad and Peter Tiňo. Adaptive metric learning vector quantization for ordinal classification. *Neural Computation*, 24(11):2825–2851, 2012. (Cited on pages 9, 30, and 75.)

[35] Jan Verwaeren, Willem Waegeman, and Bernard De Baets. Learning partial ordinal class memberships with kernel-based proportional odds models. *Computational Statistics & Data Analysis*, 56(4):928–942, 2012. (Cited on pages 9, 10, 30, 31, 59, and 68.)

[36] Peter McCullagh. Regression models for ordinal data. *Journal of the Royal Statistical Society. Series B (Methodological)*, 42(2):109–142, 1980. ISSN 00359246. (Cited on pages 69, 76, 120, and 132.)

[37] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In A.J. Smola, P.L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 115–132, Cambridge, MA, 2000. MIT Press. (Cited on pages 56, 69, and 70.)

[38] Koby Crammer and Yoram Singer. Pranking with ranking. In *Advances in Neural Information Processing Systems 14*, pages 641–647. MIT Press, 2001. (Cited on pages 67 and 72.)

[39] Wei Chu and S. Sathiya Keerthi. New approaches to support vector ordinal regression. In *In ICML'05: Proceedings of the 22nd international conference on Machine Learning*, pages 145–152, New York, NY, USA, 2005. ACM. (Cited on pages 9, 30, 71, 76, 118, 127, and 132.)

[40] Pedro A. Gutiérrez, M. Pérez-Ortiz, F. Fernandez-Navarro, J. Sánchez-Monedero, and C. Hervás-Martínez. An Experimental Study of Different Ordinal Regression Methods and Measures. In *7th International Conference on Hybrid Artificial Intelligence Systems*, pages 296–307, 2012. (Cited on pages 10, 31, and 55.)

[41] W. Chu and Z. Ghahramani. Gaussian processes for ordinal regression. *Journal of Machine Learning Research*, 6:1019–1041, 2005. (Cited on pages 11, 32, 56, 60, 72, 76, 110, 117, 130, 132, 154, and 167.)

[42] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1-3):489 – 501, 2006. ISSN 0925-2312. (Cited on pages 12, 33, and 89.)

[43] Wei-Yang Lin, Hu, Ya-Han Hu, Tsai, and Chih-Fong Tsai. Machine Learning in Financial Crisis Prediction: A Survey. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 42(4):421–436, 2011. (Cited on pages 13, 34, 148, 149, and 150.)

[44] V.N. Vapnik. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999, 1999. (Cited on pages 37, 76, 99, 120, 127, and 150.)

[45] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 2nd edition, 1998. ISBN 0132733501. (Cited on page 38.)

[46] W. S. McCulloch and W. H. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943. (Cited on pages 38 and 40.)

[47] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, August 2001. (Cited on pages 40 and 66.)

[48] S. H Lee and C. L. Hou. An art-based construction of RBF networks. *IEEE Transactions on Neural Networks*, 13(6):1308–1321, 2002. (Cited on page 40.)

[49] C. M. Bishop. Improving the generalization properties of radial basis function neural networks. *Neural Computation*, 3(4):579–581, 1991. (Cited on page 40.)

[50] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, 1989. (Cited on pages 40 and 41.)

[51] R. Durbin and D. Rumelhart. Products units: A computationally powerful and biologically plausible extension to backpropagation networks. *Neural Computation*, 1(1):133–142, 1989. (Cited on pages 40 and 41.)

[52] M. Schmitt. On the complexity of computing and learning with multiplicative neural networks. *Neural Computation*, 14:241–301, 2002. (Cited on page 40.)

[53] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. (Cited on page 41.)

[54] K. Hornik, M. Stinchcombe, and H. White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks*, 3(5):551–560, 1990. (Cited on page 41.)

[55] F. J. Martínez-Estudillo, C. Hervás-Martínez, P. A. Gutiérrez, and A. C. Martínez-Estudillo. Evolutionary product-unit neural networks classifiers. *Neurocomputing*, 72(1-2):548–561, 2008. (Cited on page 41.)

[56] A. C. Martínez-Estudillo, F. J. Martínez-Estudillo, C. Hervás-Martínez, and N. García. Evolutionary product unit based neural networks for regression. *Neural Networks*, 19(4):477–486, 2006. (Cited on page 41.)

[57] J. Park and I. W. Sandberg. Approximation and radial-basis-function networks. *Neural Computation*, 5(2):305–316, 1993. ISSN 0899-7667. (Cited on page 42.)

[58] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3): 273–297, 1995. (Cited on pages 44, 49, 76, 99, 127, and 150.)

[59] M. A. Aizerman, E. A. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. In *Automation and Remote Control,*, number 25 in Automation and Remote Control,, pages 821–837, 1964. (Cited on page 46.)

[60] B.E. Boser, I.M. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 144–152, Pittsburgh, PA, 1992. ACM Press. (Cited on page 46.)

[61] Bernhard Schölkopf, Alex J. Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5): 1299–1319, 1998. (Cited on page 47.)

[62] Sebastian Mika. *Fisher Discriminant Analysis With Kernels*. PhD thesis, Berlin, Dec. 2002. (Cited on page 47.)

[63] Volker Roth and Volker Steinhage. Nonlinear discriminant analysis using kernel functions. In *NIPS*, pages 568–574, 1999. (Cited on page 47.)

[64] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1999. (Cited on page 51.)

[65] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. (Cited on page 54.)

[66] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: an update. *Special Interest Group on Knowledge Discovery and Data Mining Explorer Newsletter*, 11: 10–18, November 2009. ISSN 1931-0145. (Cited on pages 54, 76, 154, and 167.)

[67] Anil K. Jain, Robert P.W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22 (1):4–37, 2000. (Cited on page 55.)

[68] Vladimir Cherkassky and Filip M. Mulier. *Learning from Data: Concepts, Theory, and Methods*. Wiley-Interscience, 2007. (Cited on page 55.)

[69] J. A. Anderson. Regression and ordered categorical variables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 46(1):1–30, 1984. (Cited on page 56.)

[70] R. Bender and U. Grouven. Ordinal logistic regression in medical research. *Journal of the Royal College of Physicians of London*, 31(5):546–551, 1997. (Cited on page 56.)

[71] Ralf Bender and Ulrich Grouven. Using binary logistic regression models for ordinal data with non-proportional odds. *Journal of clinical epidemiology*, 51(10): 809–816, 1998.

[72] Won Mo Jang, Sang Jun Eun, Chae-Eun Lee, and Yoon Kim. Effect of repeated public releases on cesarean section rates. *Journal of Preventive Medicine and Public Health*, 44(1):2–8, 2011.

[73] Yong Fan. Ordinal ranking for detecting mild cognitive impairment and alzheimer's disease based on multimodal neuroimages and CSF biomarkers. In Tianming Liu, Dinggang Shen, Luis Ibanez, and Xiaodong Tao, editors, *Proceedings of the First International Workshop on Multimodal Brain Image Analysis (MBIA2011)*, volume 7012 of *Lecture Notes in Computer Science*, pages 44–51. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-24445-2.

[74] Chi-Kan Chen. The classification of cancer stage microarray data. *Computer Methods and Programs in Biomedicine*, 108(3):1070–1077, December 2012.

[75] M. Perez-Ortiz, P. A. Gutierrez, C. Garcia-Alonso, L. Salvador-Carulla, J. A. Salinas-Perez, and C. Hervas-Martinez. Ordinal classification of depression spatial hot-spots of prevalence. In *11th International Conference on Intelligent Systems Design and Applications (ISDA)*, pages 1170–1175, nov. 2011. (Cited on pages 72 and 75.)

[76] María Pérez-Ortiz, Pedro Antonio Gutiérrez, César Hervás-Martínez, J. Briceño, and M. de la Mata. An ensemble approach for ordinal threshold models applied to liver transplantation. In *Proceedings of the 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 2795–2802, 2012. (Cited on pages 56 and 66.)

[77] Kuang-Yu Chang, Chu-Song Chen, and Yi-Ping Hung. Ordinal hyperplanes ranker with cost sensitivities for age estimation. In *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 585 –592, june 2011. (Cited on pages 56 and 66.)

[78] Ji Won Yoon, Stephen J. Roberts, Mathew Dyson, and John Q. Gan. Bayesian inference for an adaptive ordered probit model: An application to brain computer interfacing. *Neural Networks*, 24(7):726–734, 2011. (Cited on page 56.)

[79] Y.S. Kwon, I. Han, and K.C. Lee. Ordinal pairwise partitioning (OPP) approach to neural networks training in bond rating. *Intelligent Systems in Accounting Finance and Management*, 6(1):23–40, 1997. (Cited on pages 56 and 66.)

[80] H. Dikkers and L. Rothkrantz. Support vector machines in ordinal classification: An application to corporate credit scoring. *Neural Network World*, 15(6):491–507, 2005. (Cited on page 56.)

[81] Mark J. Mathieson. Ordinal models for neural networks. In J. Moody A.-P. N. Refenes, Y. Abu-Mostafa and A. Weigend, editors, *Proceedings of the Third International Conference on Neural Networks in the Capital Markets*, Neural Networks in Financial Engineering, pages 523–536. World Scientific, 1996. (Cited on pages 56, 70, and 75.)

[82] Minyoung Kim and Vladimir Pavlovic. Structured Output Ordinal Regression for Dynamic Facial Emotion Intensity Prediction. In Kostas Daniilidis, Petros

Maragos, and Nikos Paragios, editors, *Proceedings of the 11th European Conference on Computer Vision (ECCV 2010), Part III*, volume 6313 of *Lecture Notes in Computer Science*, pages 649–662. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-15557-4. (Cited on page 56.)

[83] O. Rudovic, V. Pavlovic, and M. Pantic. Multi-output Laplacian dynamic ordinal regression for facial expression recognition and intensity estimation. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2634–2641, 2012.

[84] Ognjen Rudovic, Vladimir Pavlovic, and Maja Pantic. Kernel Conditional Ordinal Random Fields for Temporal Segmentation of Facial Action Units. In Andrea Fusiello, Vittorio Murino, and Rita Cucchiara, editors, *Proceedings of the European Conference on Computer Vision Computer Vision (ECCV 2012). Part II*, volume 7584 of *Lecture Notes in Computer Science*, pages 260–269. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-33867-0. (Cited on page 56.)

[85] María Pérez-Ortiz, Antonio Arauzo-Azofra, César Hervás-Martínez, Laura García-Hernández, and Lorenzo Salas-Morera. A system learning user preferences for multiobjective optimization of facility layouts. In *Proceedings on the Int. Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO'12)*, 2012. (Cited on page 56.)

[86] Andrew S. Fullerton and Jun Xu. The proportional odds with partial proportionality constraints model for ordinal response variables. *Social Science Research*, 41(1):182–198, 2012. ISSN 0049-089X. (Cited on pages 56 and 70.)

[87] Tie-Yan Liu. *Learning to Rank for Information Retrieval*. Springer-Verlag, 2011. (Cited on pages 56, 58, and 59.)

[88] Chun-Wei Seah, Ivor W. Tsang, and Yew-Soon Ong. Transductive ordinal regression. *IEEE Transactionss on Neural Networks and Learning Systems*, 23(7): 1074–1086, 2012. (Cited on pages 56 and 59.)

[89] Bing-Yu Sun, Jiuyong Li, Desheng Dash Wu, Xiao-Ming Zhang, and Wen-Bo Li. Kernel discriminant learning for ordinal regression. *IEEE Transactions on Knowledge and Data Engineering*, 22(6):906–910, 2010. (Cited on pages 56, 71, 72, 76, 127, and 132.)

[90] Constantin Zopounidis and Michael Doumpos. Multicriteria classification and sorting methods: A literature review. *European Journal of Operational Research*, 138(2):229–246, 2002. (Cited on page 57.)

[91] Ricardo Sousa, Iryna Yevseyeva, Joaquim F. Pinto da Costa da Costa, and Jaime S. Cardoso. Multicriteria models for learning ordinal data: A literature review. In Xin-She Yang, editor, *Artificial Intelligence, Evolutionary Computing and Metaheuristics: In the Footsteps of Alan Turing*, pages 109–138. Springer Verlag, 2013. (Cited on page 57.)

[92] Arie Ben-David, Leon Sterling, and TriDat Tran. Adding monotonicity to learning algorithms may impair their accuracy. *Expert Systems with Applications*, 36 (3):6627–6634, 2009. (Cited on page 57.)

[93] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960. (Cited on page 57.)

[94] Shyamsundar Rajaram, Ashutosh Garg, Xiang Sean Zhou, and Thomas S. Huang. Classification Approach towards Ranking and Sorting Problems. In *Proceedings of the 14th European Conference on Machine Learning (ECML2003)*, volume 2837 of *Lecture Notes in Computer Science*, pages 301–312, 2003. (Cited on pages 58 and 74.)

[95] Shivani Agarwal and Dan Roth. Learnability of Bipartite Ranking Functions. In *Proceedings of the 18th Annual Conference on Learning Theory (COLT2005)*, volume 3559 of *Lecture Notes in Computer Science*, pages 16–31. Springer Berlin Heidelberg, 2005. (Cited on page 58.)

[96] Shyamsundar Rajaram and Shivani Agarwal. Generalization bounds for k-partite ranking. In *Proceedings of the Seventeenth Annual Conference on Neural Information Processing Systems (NIPS2005)*, pages 28–23, 2005.

[97] Johannes Fürnkranz and Eyke Hüllermeier. Preference learning: An introduction. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 1–17. Springer Berlin Heidelberg, 2011.

[98] Johannes Fürnkranz, Eyke Hüllermeier, and Stijn Vanderlooy. Binary decomposition methods for multipartite ranking. In *Proceedings of the European Conference on Machine Learning (ECML PKDD 2009), Part I*, volume 578 of *Lecture Notes in Computer Science*, pages 359–374, 2009. (Cited on page 58.)

[99] Thorsten Joachims. *Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms*. Kluwer Academic Publishers / Springer, May 2002. (Cited on page 59.)

[100] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An Efficient Boosting Algorithm for Combining Preferences. *J. of Machine Learning Research*, 4:933–969, 2003. (Cited on pages 59 and 73.)

[101] Qinghua Hu, Weiwei Pan, Lei Zhang, David Zhang, Yanping Song, Maozu Guo, and Daren Yu. Feature selection for monotonic classification. *IEEE Transactions on Fuzzy Systems*, 20(1):69–81, 2012. (Cited on page 59.)

[102] Qinghua Hu, Xunjian Che, Lei Zhang, D. Zhang, Maozu Guo, and D. Yu. Rank entropy-based decision trees for monotonic classification. *IEEE Transactions on Knowledge and Data Engineering*, 24(11):2052 –2064, 2012. ISSN 1041-4347. (Cited on pages 59 and 75.)

[103] Wojciech Kotłowski, Krzysztof Dembczyński, Salvatore Greco, and Roman Słowiński. Stochastic dominance-based rough set model for ordinal classification. *Information Sciences*, 178(21):4019–4037, 2008. ISSN 0020-0255. (Cited on page 59.)

[104] Wojciech Kotlowski and Roman Slowiński. On nonparametric ordinal classification with monotonicity constraints. *IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints):1, 2012. ISSN 1041-4347. (Cited on page 59.)

[105] R. Potharst and A. J. Feelders. Classification trees for problems with monotonicity constraints. *ACM SIGKDD Explorations Newsletter*, 4(1):1–10, jun 2002. ISSN 1931-0145. (Cited on pages 59 and 75.)

[106] Jaime Alonso, Juan José Coz, Jorge Díez, Oscar Luaces, and Antonio Bahamonde. Learning to predict one or more ranks in ordinal regression tasks. In Walter Daelemans, Bart Goethals, and Katharina Morik, editors, *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML), Part I*, volume 5211 of *Lecture Notes in Computer Science*, pages 39–54. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-87478-2. (Cited on page 59.)

[107] Dieter Devlaminck, Willem Waegeman, Bruno Bauwens, Bart Wyns, Patrick Santens, and Georges Otte. From circular ordinal regression to multilabel classification. In *Proceedings of the 2010 Workshop on Preference Learning (European Conference on Machine Learning, ECML)*, 2010. (Cited on page 59.)

[108] J. Sánchez-Monedero, P. A. Gutiérrez, F. Fernández-Navarro, and C. Hervás-Martínez. Weighting efficient accuracy and minimum sensitivity for evolving multi-class classifiers. *Neural Processing Letters*, 34(2):101–116, 2011. (Cited on pages 60, 79, and 110.)

[109] A Agresti. *Analysis of ordinal categorical data*. New York: Wiley, 1984. (Cited on page 60.)

[110] Willem Waegeman and Bernard De Baets. A Survey on ROC-based Ordinal Regression. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 127–154. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-14125-6. (Cited on page 60.)

[111] C. Spearman. The proof and measurement of association between two things. *American Journal of Psychology*, 15:72–101, 1904. (Cited on page 61.)

[112] J. L. Fleiss, J. Cohen, and B. S. Everitt. Large sample standard errors of kappa and weighted kappa. *Psychological Bulletin*, 72(5):323–327, 1969. (Cited on page 61.)

[113] M. G. Kendall. *Rank Correlation Methods*. New York: Hafner Press, 3rd edition, 1962. (Cited on page 61.)

[114] Jaime S. Cardoso and Ricardo Sousa. Measuring the performance of ordinal classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(8):1173–1195, 2011. (Cited on page 61.)

[115] Vicenç Torra, Josep Domingo-Ferrer, Josep M. Mateo-Sanz, and Michael Ng. Regression for ordinal variables without underlying continuous variables. *Information Sciences*, 176(4):465–474, 2006. (Cited on page 63.)

[116] A.J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004. (Cited on page 63.)

[117] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Data Management Systems. Morgan Kaufmann (Elsevier), 2nd edition, 2005. ISBN 0120884070. (Cited on pages 63, 75, and 155.)

[118] Stefan Kramer, Gerhard Widmer, Bernhard Pfahringer, and Michael De Groeve. Prediction of ordinal classes using regression trees. *Fundamenta Informaticae*, 47 (1-2):1–13, 2001. (Cited on page 63.)

[119] Edward F. Harrington. Online ranking/collaborative filtering using the perceptron algorithm. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML2003)*, 2003. (Cited on pages 63 and 72.)

[120] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971. (Cited on page 63.)

[121] Alan Agresti. *Analysis of ordinal categorical data*. Wiley Series in Probability and Statistics. Wiley, 2010. (Cited on page 63.)

[122] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multi-class support vector machines. *IEEE Transaction on Neural Networks*, 13(2):415–425, 2002. (Cited on pages 63 and 76.)

[123] Han-Hsing Tu and Hsuan-Tien Lin. One-sided support vector regression for multiclass cost-sensitive classification. In *Proceedings of the Twenty-Seventh International Conference on Machine learning (ICML2010)*, pages 49–56, 2010. (Cited on page 63.)

[124] Zhi-Hua Zhou and Xu-Ying Liu. On multi-class cost-sensitive learning. *Computational Intelligence*, 26(3):232–257, 2010. (Cited on page 63.)

[125] Alina Beygelzimer, Varsha Dani, Tom Hayesand John Langford, and Bianca Zadrozny. Error limiting reductions between classification tasks. In *Proceedings of the 22nd international conference on Machine learning (ICML05)*, pages 49–56, 2005. (Cited on page 63.)

[126] Erin L. Allwein, Robert E. Schapire, and Yoram Singer. Reducing multiclass to binary: a unifying approach for margin classifiers. *J. of Machine Learning Research*, 1:113–141, 2001. ISSN 1532-4435. (Cited on pages 65, 67, and 108.)

[127] Ulrich Paquet, Sean Holden, and Andrew Naish-Guzman. Bayesian hierarchical ordinal regression. In *Proceedings of the International Conferences on Artificial Neural Networks (ICANN2005)*, volume 3697 of *Lecture Notes in Computer Science*, 2005. (Cited on page 66.)

[128] Hong Wu, Hanqing Lu, and Songde Ma. A practical SVM-based algorithm for ordinal regression in image retrieval. In *Proceedings of the eleventh ACM international conference on Multimedia (Multimedia2003)*, pages 612–621, 2003. (Cited on page 66.)

[129] Mario Costa. Probabilistic interpretation of feedforward network outputs, with relationships to statistical prediction of ordinal quantities. *International Journal of Neural Systems*, 7(5):627–638, 1996. (Cited on pages 67 and 68.)

[130] Jianlin Cheng, Zheng Wang, and Gianluca Pollastri. A neural network approach to ordinal regression. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN2008, IEEE World Congress on Computational Intelligence)*, pages 1279–1284. IEEE Press, 2008. (Cited on pages 67 and 110.)

[131] Guang-Bin Huang, Hongming Zhou, Xiaojian Ding, and Rui Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 42(2):513–529, 2012. (Cited on pages 67 and 111.)

[132] Wan-Yu Deng, Qing-Hua Zheng, Shiguo Lian, Lin Chen, and Xin Wang. Ordinal extreme learning machine. *Neurocomputing*, 74(1–3):447–456, 2010. (Cited on pages 67 and 107.)

[133] Joaquim F. Pinto da Costa and Jaime S. Cardoso. Classification of ordinal data using neural networks. In João Gama, Rui Camacho, PavelB. Brazdil, AlípioMário Jorge, and Luís Torgo, editors, *Proceedings of the 16th European Conference on Machine Learning (ECML 2005)*, volume 3720 of *Lecture Notes in Computer Science*, pages 690–697. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-29243-2. (Cited on pages 68 and 75.)

[134] Gerhard Tutz and Wolfgang Hennevogl. Random effects in ordinal regression models. *Computational Statistics & Data Analysis*, 22(5):537 – 557, 1996. ISSN 0167-9473. (Cited on page 68.)

[135] Willem Waegeman, Bernard De Baets, and Luc Boullart. Roc analysis in ordinal regression learning. *Pattern Recognition Letters*, 29(1):1–9, 2008. (Cited on page 68.)

[136] A. Agresti. *Categorical Data Analysis*. John Wiley and Sons, 2 edition, 2002. (Cited on pages 69 and 74.)

[137] Mark J. Mathieson. Ordered classes and incomplete examples in classification. In Thomas Petsche Michael C. Mozer, Michael I. Jordan, editor, *Proceedings of the 1996 Conference on Neural Information Processing Systems (NIPS)*, volume 9 of *Advances in Neural Information Processing Systems*, pages 550–556, 1999. (Cited on pages 69 and 70.)

[138] P. McCullagh and John A. Nelder. *Generalized Linear Models*. Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, 2nd edition, 1989. (Cited on pages 69, 70, and 110.)

[139] Richard Williams. Generalized ordered logit/partial proportional odds models for ordinal dependent variables. *Stata Journal*, 6(1):58–82, March 2006. (Cited on page 70.)

[140] Bercedis Peterson and Jr Harrell, Frank E. Partial proportional odds models for ordinal response variables. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 39(2):205–217, 1990. ISSN 00359254. (Cited on page 70.)

[141] G. Tutz. Generalized semiparametrically structured ordinal models. *Biometrics*, 59(2):263–273, 2003. (Cited on page 70.)

[142] Manuel Dorado-Moreno, Pedro Antonio Gutiérrez, and César Hervás-Martínez. Ordinal classification using hybrid artificial neural networks with projection and kernel basis functions. In *7th International Conference on Hybrid Artificial Intelligence Systems (HAIS2012)*, pages 319–330, 2012. (Cited on page 70.)

[143] T. Van Gestel, B. Baesens, P. Van Dijcke, J. Garcia, J.A.K. Suykens, and J. Vanthienen. A process model to develop an internal rating system: Sovereign credit ratings. *Decision Support Systems*, 42(2):1131–1151, 2006. (Cited on pages 70 and 150.)

[144] Amnon Shashua and Anat Levin. Ranking with large margin principle: Two approaches. In *Proceedings of the Seventeenth Annual Conference on Neural Information Processing Systems (NIPS2003)*, number 16 in Advances in Neural Information Processing Systems, pages 937–944. MIT Press, 2003. (Cited on page 70.)

[145] S. K. Shevade and Wei Chu. Minimum enclosing spheres formulations for support vector ordinal regression. In *Proceedings of the Sixth International Conference on Data Mining (ICDM '06)*, pages 1054–1058, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2701-9. (Cited on page 71.)

[146] Shivani Agarwal. Generalization bounds for some ordinal regression algorithms. In *Algorithmic Learning Theory*, volume 5254 of *Lecture Notes in Artificial Intelligence (Lecture Notes in Computer Science)*, pages 7–21. Springer-Verlag Berlin Heidelberg, 2008. (Cited on page 71.)

[147] T. Xu, Y. Zhang, and W. Gao. Generalization bounds for ordinal regression algorithms via strong and weak stability. *Energy Procedia*, 13:3471–3478, 2011. (Cited on page 71.)

[148] Emilio Carrizosa and Belen Martin-Barragan. Maximizing upgrading and downgrading margins for ordinal regression. *Mathematical Methods of Operations Research*, 74(3):381–407, 2011. (Cited on page 71.)

[149] B. Zhao, F. Wang, and C. Zhang. Block-quantized support vector ordinal regression. *IEEE Transactions on Neural Networks*, 20(5):882–890, 2009. (Cited on page 71.)

[150] Bin Gu, Jian-Dong Wang, and Tao Li. Ordinal-class core vector machine. *Journal of Computer Science and Technology*, 25(4):699–708, 2010. (Cited on page 71.)

[151] Ivor W. Tsang, James T. Kwok, and Pak-Ming Cheung. Core vector machines: Fast SVM training on very large data sets. *J. of Machine Learning Research*, 6: 363–392, 2005. (Cited on page 71.)

[152] Yang Liu, Yan Liu, Shenghua Zhong, and Keith C.C. Chan. Semi-supervised manifold ordinal regression for image ranking. In *Proceedings of the 19th ACM international conference on Multimedia (ACM MM2011)*, pages 1393–1396, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0616-4. (Cited on page 72.)

[153] Yang Liu, Yan Liu, and Keith C. C. Chan. Ordinal regression via manifold learning. In Wolfram Burgard and Dan Roth, editors, *Proceedings of the 25th*

AAAI Conference on Artificial Intelligence (AAAI'11), pages 398–403. AAAI Press, 2011. (Cited on page 72.)

[154] Yang Liu, Yan Liu, Keith C. C. Chan, and Jing Zhang. Neighborhood preserving ordinal regression. In *Proceedings of the 4th International Conference on Internet Multimedia Computing and Service (ICIMCS12)*, pages 119–122, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1600-2. (Cited on page 72.)

[155] Justin Basilico and Thomas Hofmann. Unifying collaborative and content-based filtering. In *Proceedings of the twenty-first international conference on machine learning (ICML)*, 2004. (Cited on page 72.)

[156] Wei Chu and Zoubin Ghahramani. Preference learning with gaussian processes. In *Proceedings of the 22nd international conference on Machine learning (ICML2005)*, pages 137–144, 2005. (Cited on page 73.)

[157] Xiao Chang, Qinghua Zheng, and Peng Lin. Ordinal regression with sparse bayesian. In *Proceedings of the 5th International Conference on Intelligent Computing (ICIC2009)*, volume 5755 of *Lecture Notes in Computer Science*, pages 591–599, 2009. (Cited on page 73.)

[158] Hsuan-Tien Lin and Ling Li. Large-margin thresholded ensembles for ordinal regression: Theory and practice. In José L. Balcázar, Philip M. Long, and Frank Stephan, editors, *Proceeding of the 17th Algorithmic Learning Theory International Conference (ALT2006)*, volume 4264 of *Lecture Notes in Artificial Intelligence (LNAI)*, pages 319–333. Springer-Verlag, October 2006. (Cited on pages 73 and 74.)

[159] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004. (Cited on page 73.)

[160] Hsuan-Tien Lin and Ling Li. Combining ordinal preferences by boosting. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, pages 69–83, 2009. (Cited on pages 73 and 74.)

[161] Fen Xia, Liang Zhou, Yanwu Yang, and Wensheng Zhang. Ordinal regression as multiclass classification. *International Journal of Intelligent Control and Systems*, 12(3):230–236, 2007. (Cited on page 74.)

[162] Joaquim F. Pinto da Costa, Ricardo Sousa, and Jaime S. Cardoso. An all-at-once unimodal SVM approach for ordinal classification. In *Proceedings of the Ninth International Conference on Machine Learning and Applications (ICMLA2010)*, pages 59–64. IEEE Computer Society Press, 2010. (Cited on page 75.)

[163] J.S. Cardoso and R. Sousa. Classification models with global constraints for ordinal data. In *Proceedings of the Ninth International Conference on Machine Learning and Applications (ICMLA2010)*, pages 71 –77, 2010. (Cited on page 75.)

[164] Salvador Garcia, Joaquin Derrac, Jose Cano, and Francisco Herrera. Prototype selection for nearest neighbor classification: Taxonomy and empirical

study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):417–435, 2012. ISSN 0162-8828. (Cited on page 75.)

[165] R. Sousa and J.S. Cardoso. Ensemble of decision trees with global constraints for ordinal classification. In *Proceedings of the 11th International Conference on Intelligent Systems Design and Applications (ISDA2011)*, pages 1164–1169, 2011. (Cited on page 75.)

[166] Shereen Fouad and Peter Tino. Prototype based modelling for ordinal classification. In Hujun Yin, JoséA.F. Costa, and Guilherme Barreto, editors, *Proceedings of the 13th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL2012)*, volume 7435 of *Lecture Notes in Computer Science*, pages 208–215. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-32638-7. (Cited on page 75.)

[167] D. J. C. Mackay. Bayesian methods for backpropagation networks. In E. Domany, J. L. van Hemmen, and K. Schulten, editors, *Models of Neural Networks III*, chapter 6, pages 211–254. Springer-Verlag, New York, 1994. (Cited on page 76.)

[168] Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, New York, Secaucus, NJ, USA, 1996. ISBN 0387947248. (Cited on page 76.)

[169] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2:27:1–27:27, May 2011. ISSN 2157-6904. (Cited on pages 76, 100, 118, 132, and 154.)

[170] J. Sánchez-Monedero, C. Hervás-Martínez, F. Martínez-Estudillo, Mariano Ruz, M. Moreno, and M. Cruz-Ramírez. Evolutionary learning using a sensitivity-accuracy approach for classification. In *Hybrid Artificial Intelligence Systems*, volume 6077 of *Lecture Notes in Computer Science*, pages 288–295. Springer Berlin / Heidelberg, 2010. (Cited on page 79.)

[171] J. Sánchez-Monedero, C. Hervás-Martínez, P.A. Gutiérrez, M. Carbonero-Ruz, M. C. Ramírez-Moreno, and M. Cruz-Ramírez. Evaluating the Performance of Evolutionary Extreme Learning Machines by a Combination of Sensitivity and Accuracy Measures. *Neural Network World*, 20:899–912, 2010. (Cited on page 79.)

[172] X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999. (Cited on page 80.)

[173] Guang-Bin Huang, Lei Chen, and Chee-Kheong Siew. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Transactions on Neural Networks*, 17(4):879–892, July 2006. ISSN 1045-9227. (Cited on pages 80 and 89.)

[174] R. Penrose. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51:406–413, 6 1955. ISSN 1469-8064. (Cited on pages 80 and 90.)

[175] E. H. Moore. On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society*, 26(9):394–395, 1920. (Cited on pages 80 and 90.)

[176] Qin-Yu Zhu, A.K. Qin, P.N. Suganthan, and Guang-Bin Huang. Evolutionary extreme learning machine. *Pattern Recognition*, 38(10):1759 – 1763, 2005. ISSN 0031-3203. (Cited on pages 80, 90, 92, 93, 94, and 111.)

[177] Rainer Storn and Kenneth Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997. ISSN 0925-5001. (Cited on pages 80 and 90.)

[178] N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis Journal*, 6(5), 2002. (Cited on pages 81 and 87.)

[179] J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano. Experimental perspectives on learning from imbalanced data. In *Proceedings of the 24th international conference on Machine learning (ICML'07)*, volume 227, pages 935–942, 2007.

[180] A. Fernández, S. García, M.J. del Jesus, and F. Herrera. A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. *Fuzzy Sets and Systems*, 159(18):2378–2398, 2008. (Cited on pages 83 and 85.)

[181] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 9(21):1263–1284, 2009. (Cited on page 81.)

[182] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16: 321–357, 2002. (Cited on pages 81 and 87.)

[183] Francisco Fernández-Navarro, César Hervás-Martínez, and Pedro Antonio Gutiérrez. A dynamic over-sampling procedure based on sensitivity for multi-class problems. *Pattern Recognition*, 44(8):1821–1833, 2011. ISSN 0031-3203. (Cited on pages 81, 85, and 87.)

[184] M. Cruz-Ramírez, C. Hervás-Martínez, P.A. Gutiérrez, M. Pérez-Ortiz, J. Briceño, and M. de la Mata. Memetic Pareto differential evolutionary neural network used to solve an unbalanced liver transplantation problem. *Soft Computing*, 17(2):275–284, 2013. ISSN 1432-7643. (Cited on page 81.)

[185] M. Kubat, R. C. Holte, and S. Matwin. Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30(2-3):195–215, 1998. (Cited on pages 81, 83, and 85.)

[186] T. Fawcett and F. Provost. Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 1(3):291–316, 1997. (Cited on page 81.)

[187] Kazuo Ezawa, Moninder Singh, and Steven W. Norton. Learning goal oriented bayesian networks for telecommunications risk management. In *In Proceedings of the 13th International Conference on Machine Learning*, pages 139–147. Morgan Kaufmann, 1996. (Cited on pages 81 and 87.)

[188] F. Fernández-Navarro, Antonio Valero, C. Hervás-Martínez, P. A. Gutiérrez, Rosa M. García-Gimeno, and Gonzalo Zurera-Cosano. Development of a multi-classification neural network model to determine the microbial growth/no growth interface. *International Journal of Food Microbiology*, 141(3):203–212, 2010. (Cited on page 81.)

[189] C. Cardie and N. Howe. Improving minority class prediction using case-specific feature weights. *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 57–65, 1997. (Cited on page 81.)

[190] R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard. Class imbalances versus class overlapping: An analysis of a learning system behavior. In *MICAI 2004 : advances in artificial intelligence, Lecture notes in Computer Science*, volume 2972, pages 312–321, 2004. (Cited on page 81.)

[191] Yi L. Murphey, Hong Guo, and Lee A. Feldkamp. Neural learning from unbalanced data. *Applied Intelligence*, 21:117–128, 2004. (Cited on page 81.)

[192] A. Fernández, S. García, and F. Herrera. Addressing the Classification with Imbalanced Data: Open Problems and New Challenges on Class Distribution. In Emilio Corchado, Marek Kurzyński, and Michał Woźniak, editors, *Hybrid Artificial Intelligent Systems*, volume 6678 of *Lecture Notes in Computer Science*, pages 1–10. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-21218-5. (Cited on pages 81 and 87.)

[193] Son Lam Phung, Abdesselam Bouzerdoum, and Giang Hoang Nguyen. *Pattern recognition. Chapter "Learning pattern classification tasks with imbalanced data sets"*. InTech, 2009. ISBN 978-953-307-014-8. URL http://www.intechopen.com/books/pattern-recognition/learning-pattern-classification-tasks-with-imbalanced-data-sets. Peng-Yeng Yin, Ed. (Cited on page 81.)

[194] G. M. Weiss. Mining with rarity: A unifying framework. *ACM SIGKDD Explorations Newsletter*, pages 67–119, 2004. (Cited on pages 81, 83, and 87.)

[195] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27: 861–874, 2006. (Cited on pages 82 and 84.)

[196] Miroslav Kubat and Stan Matwin. Addressing the curse of imbalanced training sets: one-sided selection. In *Proceedings of the 14th International Conference on Machine Learning*, pages 179–186. Morgan Kaufmann, 1997. (Cited on page 83.)

[197] R. Barandela, J.S. Sánchez, V. García, and E. Rangel. Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851, 2003. ISSN 0031-3203. (Cited on pages 83 and 85.)

[198] D.J. Hand and R.J. Till. A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems. *Machine Learning*, 45(2):171–186, 2001. (Cited on page 84.)

[199] J.S Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. *Neurocomputing: Algorithms, Architectures and Applications*, pages 227–236, 1990. (Cited on pages 84, 91, and 108.)

[200] A.P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997. (Cited on page 84.)

[201] Jin 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. ISSN 1041-4347. (Cited on page 84.)

[202] Ashwin Srinivasan. Note on the location of optimal classifiers in n-dimensional roc space. Technical report, Computing Laboratory, 1999. (Cited on page 84.)

[203] C. Ferri, J. Hernández-orallo, and M.A. Salido. Volume under the ROC surface for multi-class problems. exact computation and evaluation of approximations. In *Proc. of 14th European Conference on Machine Learning*, pages 108–120, 2003. (Cited on page 84.)

[204] Richard M. Everson and Jonathan E. Fieldsend. Multi-class ROC analysis from a multi-objective optimisation perspective. *Pattern Recognition Letters*, 27(8):918–927, 2006. (Cited on page 84.)

[205] Foster Provost and Tom Fawcett. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *In Proc. of the 3rd International Conference on Knowledge Discovery and Data Mining*, pages 43–48, 1997. (Cited on page 85.)

[206] F. J. Martínez-Estudillo, P. A. Gutiérrez, C. Hervás-Martínez, and J. C. Fernández. Evolutionary learning by a sensitivity-accuracy approach for multi-class problems. In *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (CEC'08)*, pages 1581–1588, Hong Kong, China, 2008. IEEE Press. (Cited on pages 85, 86, and 90.)

[207] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378, 2007. (Cited on page 87.)

[208] Zhi-Hua Zhou and Xu-Ying Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):63–77, 2006. ISSN 1041-4347. (Cited on page 87.)

[209] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations*, 6(1):20–29, 2004. (Cited on page 87.)

[210] Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi, editors, *Machine Learning: ECML*

2004, volume 3201 of *Lecture Notes in Computer Science*, pages 39–50. Springer Berlin Heidelberg, 2004. ISBN 978-3-540-23105-9. (Cited on page 87.)

[211] B. Raskutti and A. Kowalczyk. Extreme re-balancing for svms: A case study. *SIGKDD Explorations*, 6(1):60–69, 2004. (Cited on page 87.)

[212] Andrew K. C. Wong and Yang Wang. High-order pattern discovery from discrete-valued data. *IEEE Transactions on Knowledge and Data Engineering*, 9 (6):877–893, 1997. (Cited on page 87.)

[213] H. A. Abbass, R. Sarker, and C. Newton. PDE: a Pareto-frontier differential evolution approach formulti-objective optimization problems. In *Proceedings of the 2001 Congress on Evolutionary Computation*, volume 2, Seoul, South Korea, 2001. (Cited on pages 88 and 99.)

[214] Hussein A. Abbass. Speeding up backpropagation using multiobjective evolutionary algorithms. *Neural Computation*, 15(11):2705–2726, 2003. ISSN 0899-7667. (Cited on pages 88 and 97.)

[215] Kalyanmoy Deb, Amrit Pratab, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA2. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002. (Cited on page 88.)

[216] Christian Igel and Michael Hüsken. Empirical evaluation of the improved rprop learning algorithms. *Neurocomputing*, 50(6):105–123, 2003. (Cited on pages 88 and 99.)

[217] J.C. Fernández, C. Hervás, F.J. Martínez-Estudillo, and P.A. Gutiérrez. Memetic Pareto Evolutionary Artificial Neural Networks to determine growth/no-growth in predictive microbiology. *Applied Soft Computing*, 11(1):534 – 550, 2011. ISSN 1568-4946. (Cited on page 88.)

[218] Carlos A. Coello. An updated survey of ga-based multiobjective optimization techniques. *ACM Computer Surveys*, 32(2):109–143, 2000. ISSN 0360-0300. (Cited on pages 88 and 97.)

[219] C. Coello. An updated survey of evolutionary multiobjective optimization techniques: State of the art and future trends. In P.J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, editors, *Proceedings of the Congress on Evolutionary Computation*, volume 1, pages 3–13. IEEE Press, 1999. (Cited on page 88.)

[220] P. B. Wilson and M. D. Macleod. Low implementation cost iir digital filter design using genetic algorithms. In *Workshop on Natural Algorithms in Signal Processing*, Chelmsford, Essex, UK, 1993. (Cited on page 88.)

[221] Runxuan Zhang, Guang-Bin Huang, N. Sundararajan, and P. Saratchandran. Multicategory classification using an extreme learning machine for microarray gene expression cancer diagnosis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(3):485–495, 2007. ISSN 1545-5963. (Cited on page 89.)

[222] J. Sanchez-Monedero, M. Cruz-Ramirez, F. Fernandez-Navarro, J.C. Fernandez, P.A. Gutierrez, and C. Hervas-Martinez. On the suitability of extreme learning machine for gene classification using feature selection. In *Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on*, pages 507–512, 2010. (Cited on page 89.)

[223] Zhan-Li Sun, Tsan-Ming Choi, Kin-Fan Au, and Yong Yu. Sales forecasting using extreme learning machine with applications in fashion retailing. *Decision Support Systems*, 46(1):411 – 419, 2008. ISSN 0167-9236. (Cited on page 89.)

[224] Wanyu Deng and Lin Chen. Color image watermarking using regularized extreme learning machine. *Neural Network World*, 20(3):317–330, 2010. (Cited on page 89.)

[225] Lei Chen, LiFeng Zhou, and Hung Pung. Universal Approximation and QoS Violation Application of Extreme Learning Machine. *Neural Processing Letters*, 28:81–95, 2008. ISSN 1370-4621. (Cited on page 89.)

[226] Jarmo Ilonen, Joni-Kristian Kamarainen, and Jouni Lampinen. Differential evolution training algorithm for feed-forward neural networks. *Neural Processing Letters*, 17(1):93–105, 2003. ISSN 1370-4621. (Cited on pages 90 and 98.)

[227] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse. OP-ELM: Optimally Pruned Extreme Learning Machine. *IEEE Transactions on Neural Networks*, 21(1):158–162, December 2010. (Cited on pages 90 and 99.)

[228] L.P. Wang and C.R. Wan. Comments on "the extreme learning machine". *IEEE Transactions on Neural Networks*, 19(8):1494–1495, 2008. (Cited on page 90.)

[229] G.-B. Huang. Reply to "comments on "the extreme learning machine"". *IEEE Transactions on Neural Networks*, 19(8):1495–1496, 2008. cited By (since 1996)4. (Cited on page 90.)

[230] Guang-Bin Huang, DianHui Wang, and Yuan Lan. Extreme learning machines: a survey. *International Journal of Machine Learning and Cybernetics*, 2(2):107–122, 2011. ISSN 1868-8071. (Cited on page 90.)

[231] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007. URL http://www.ics.uci.edu/~mlearn/MLRepository.html. (Cited on pages 95, 98, 110, 115, and 131.)

[232] César Hervás-Martínez, Manuel Silva, Pedro A. Gutiérrez, and Antonio Serrano. Multilogistic regression by evolutionary neural network as a classification tool to discriminate highly overlapping signals: Qualitative investigation of volatile organic compounds in polluted waters by using headspace-mass spectrometric analysis. *Chemometrics and Intelligent Laboratory Systems*, 92(2): 179–185, 2008. (Cited on pages 95 and 97.)

[233] L. Prechelt. PROBEN1: A set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, Fakultät für Informatik (Universität Karlsruhe), 1994. (Cited on page 95.)

[234] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1994. (Cited on page 97.)

[235] Sebastián Ventura, Cristóbal Romero, Amelia Zafra, José Delgado, and César Hervás. JCLEC: a Java framework for evolutionary computation. *Soft Computing*, 12(4):381–392, February 2008. (Cited on page 97.)

[236] M. Cruz-Ramírez, J. Sánchez-Monedero, F. Fernández-Navarro, J. Fernández, and C. Hervás-Martínez. Memetic pareto differential evolutionary artificial neural networks to determine growth multi-classes in predictive microbiology. *Evolutionary Intelligence*, pages 1–13, 2010. ISSN 1864-5909. (Cited on page 99.)

[237] H. A. Abbass. A memetic pareto evolutionary approach to artificial neural networks. In M. Brooks, D. Corbet, and M. Stumptner, editors, *AI2001*, pages 1–12. LNAI 2256, Springer-Verlag, 2001. (Cited on page 99.)

[238] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *Proceedings of the 1993 IEEE International Conference on Neural Networks*, pages 586–591, San Francisco, CA, 1993. (Cited on page 99.)

[239] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. (Cited on pages 100, 118, and 132.)

[240] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, December 2006. ISSN 1532-4435. (Cited on pages 101, 102, 112, 137, 138, and 170.)

[241] J. Sánchez-Monedero, P. A. Gutiérrez, P. Tiňo, and C. Hervás-Martínez. Exploitation of pairwise class distances for ordinal classification. *Neural Computation*, Accepted:MIT Press, 2013. (Cited on page 105.)

[242] J. Sánchez-Monedero, P.A. Gutiérrez, M. Pérez-Ortiz, and C. Hervás-Martínez. An n-spheres based synthetic data generator for supervised classification. In *International Work-Conference on Artificial Neural Networks (IWANN)*, volume 7902 of *Lecture Notes in Computer Science*, pages 613–621. Springer-Verlag Berlin Heidelberg, 2013. (Cited on pages 126 and 189.)

[243] J. Sánchez-Monedero, M. Carbonero-Ruz, D. Becerra-Alonso, F. Martínez-Estudillo, P.A. Gutiérrez, and C. Hervás-Martínez. Numerical variable reconstruction from ordinal categories based on probability distributions. In *Proceedings of the 11th International Conference on Intelligent Systems Design andApplications (ISDA 2011)*, pages 1182–1187, Cordoba, Spain, Spain, nov 2011. (Cited on page 105.)

[244] PASCAL. Pascal (Pattern Analysis, Statistical Modelling and Computational Learning) machine learning benchmarks repository, 2011. URL `http://mldata.org/`. `http://mldata.org/`. (Cited on page 110.)

[245] Luc Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, New York, 1986. (Cited on page 115.)

[246] Dr. Soeren Sonnenburg. Machine Learning Data Set Repository, 2011. URL http://mldata.org/. (Cited on pages 115 and 131.)

[247] Milton Friedman. A comparison of alternative tests of significance for the problem of m rankings. *Annals of Mathematical Statistics*, 11(1):86–92, 1940. (Cited on pages 118, 137, and 169.)

[248] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*. The MIT Press, 1st edition, December 2001. ISBN 0262194759. (Cited on page 120.)

[249] Núria Macià. *Data complexity in supervised learning: A far-reaching implication*. PhD thesis, La Salle – Universitat Ramon Llull, October 2011. (Cited on page 126.)

[250] Núria Macià, Ester Bernadó-Mansilla, Albert Orriols-Puig, and Tin Kam Ho. Learner excellence biased by data set selection: A case for data characterisation and artificial data sets. *Pattern Recognition*, 46(3):1054 – 1066, 2013. ISSN 0031-3203. (Cited on page 126.)

[251] Janick V. Frasch, Aleksander Lodwich, Faisal Shafait, and Thomas M. Breuel. A bayes-true data generator for evaluation of supervised and unsupervised learning methods. *Pattern Recogn. Lett.*, 32(11):1523–1531, August 2011. ISSN 0167-8655. (Cited on page 126.)

[252] Yaling Pei and Osmar Zaïane. A synthetic data generator for clustering and outlier analysis. Technical report, Computing Science Department. University of Alberta, 2006. (Cited on page 126.)

[253] Free Software Foundation. GNU General Public License, June 2007. URL http://www.gnu.org/licenses/gpl.html. Version 3. (Cited on pages 127, 130, and 189.)

[254] House of Lords. *Sovereign credit ratings: shooting the messenger?, 21st report of session 2010-12, report*. House of Lords papers. Stationery Office, 2011. ISBN 9780108473722. Great Britain Parliament. European Union Committee. (Cited on pages 147 and 148.)

[255] Emawtee Bissoondoyal-Bheenick. An analysis of the determinants of sovereign ratings. *Global Finance Journal*, 15(3):251–280, 2005. (Cited on pages 148, 150, and 152.)

[256] Jakob De Haan and Fabian Amtenbrink. Credit rating agencies. *SSRN eLibrary*, DNB Working Paper No. 278, January 2011. (Cited on page 148.)

[257] Rabah Arezki, Bertrand Candelon, and Amadou Sy. Sovereign rating news and financial markets spillovers: Evidence from the european debt crisis. Cesifo working paper series, CESifo Group Munich, 2011. (Cited on page 148.)

[258] Edward I. Altman and Herbert A. Rijken. Toward a bottom-up approach to assessing sovereign default risk. *Journal of Applied Corporate Finance*, 23(1):20–31, 2011. ISSN 1745-6622. (Cited on page 148.)

[259] Paula Hill, Robert Brooks, and Robert Faff. Variations in sovereign credit quality assessments across rating agencies. *Journal of Banking & Finance*, 34(6):1327–1343, 2010. ISSN 0378-4266. (Cited on pages 148 and 152.)

[260] Richard Martin Cantor and Frank Packer. Determinants and impact of sovereign credit ratings. *Economic Policy Review*, 2(2):37–53, October 1996. (Cited on pages 148 and 152.)

[261] Rasha Al-Sakka and Owain ap Gwilym. Split sovereign ratings and rating migrations in emerging economies. *Emerging Markets Review*, 11(2):79 – 97, 2010. (Cited on pages 148 and 158.)

[262] P. Ravi Kumar and V. Ravi. Bankruptcy prediction in banks and firms via statistical and intelligent techniques - a review. *European Journal of Operational Research*, 180(1):1–28, 2007. (Cited on pages 148 and 149.)

[263] Verikas, A., Kalsyte, Z., Bacauskiene, M., Gelzinis, and A. Hybrid and ensemble-based soft computing techniques in bankruptcy prediction: A survey. *Soft Computing*, 14(9):995–1010, 2010.

[264] Yi Peng, Guoxun Wang, Gang Kou, and Yong Shi. An empirical study of classification algorithm evaluation for financial risk prediction. *Applied Soft Computing*, 11(2):2906–2915, 2011. ISSN 1568-4946. (Cited on pages 148 and 149.)

[265] Julia A. Bennell, David Crabbe, Stephen Thomas, and Owain ap Gwilym. Modelling sovereign credit ratings: Neural networks versus ordered probit. *Expert Systems with Applications*, 30(3):415 – 425, 2006. (Cited on page 150.)

[266] Charles Jr. Frank and William R. Cline. Measurement of debt servicing capacity: An application of discriminant analysis. *Journal of International Economics*, 1(3): 327–344, August 1971. (Cited on page 150.)

[267] Nicholas Sargen. Economic indicators and country risk appraisal. *Economic Review*, 1:19–35, 1977.

[268] B. Abassi and R. J. Taffler. *Country Risk: A Model of Economic Performance Related to debt Servicing Capacity*. Number 36 in Working paper series (City University. Business School). London, 1982. (Cited on page 150.)

[269] Edward B. Deakin. A discriminant analysis of predictors of business failure. *Journal of Accounting Research*, 10(1), 1972. (Cited on page 150.)

[270] Zan Huang, Hsinchun Chen, Chia-Jung Hsu, Wun-Hwa Chen, and Soushan Wu. Credit rating analysis with support vector machines and neural networks: a market comparative study. *Decision Support Systems*, 37:543–558, 2004. (Cited on page 150.)

[271] Yen-Ting Hu, Rudiger Kiesel, and William Perraudin. The estimation of transition matrices for sovereign credit ratings. *Journal of Banking and Finance*, 26(7): 1383–1406, 2002. (Cited on page 150.)

[272] Craig A. Depken, Courtney LaFountain, and Roger Butters. Corruption and creditworthiness: Evidence from sovereign credit ratings. Working Papers 0601, University of Texas at Arlington, Department of Economics, April 2006.

[273] Nada Mora. Sovereign credit ratings: Guilty beyond reasonable doubt? *Journal of Banking & Finance*, 30(7):2041–2062, July 2006.

[274] António Afonso, Pedro Gomes, and Philipp Rother. Ordered response models for sovereign debt ratings. *Applied Economics Letters*, 16(8):769–773, 2009. (Cited on page 150.)

[275] J.C. Cosset and J. Roy. Predicting country risk ratings using artificial neural networks. In *Advances in Artificial Intelligence in Economics, Finance, and Management*, volume 1, pages 141–157, 1994. (Cited on page 150.)

[276] J.C.B. Cooper. Artificial neural networks versus multivariate statistics: An application from economics. *Journal of Applied Statistics*, 26(8):909–921, 1999. (Cited on page 150.)

[277] Juliana Yim and Heather Mitchell. Comparison of country risk models: hybrid neural networks, logit models, discriminant analysis and cluster techniques. *Expert Systems with Applications*, 28(1):137–148, 2005. (Cited on page 150.)

[278] Francis E. H. Tay and Lijuan Cao. Application of support vector machines in financial time series forecasting. *Omega: The International Journal of Management Science*, 29(4):309–317, 2001. (Cited on page 150.)

[279] Y.-C. Lee. Application of support vector machines to corporate credit rating prediction. *Expert Systems with Applications*, 33(1):67–74, 2007. (Cited on page 150.)

[280] Ligang Zhou, Kin Keung Lai, and Jerome Yen. Credit Scoring Models with AUC Maximization based on Weighted SVM. *International Journal of Information Technology & Decision Making*, 8(04):677–696, 2009. (Cited on page 150.)

[281] Lean Yu, Shouyang Wang, and Jie Cao. A modified least squares support vector machine classifier with application to credit risk analysis. *International Journal of Information Technology & Decision Making*, 08(04):697–710, 2009. (Cited on page 150.)

[282] Arindam Chaudhuri and Kajal De. Fuzzy Support Vector Machine for bankruptcy prediction. *Applied Soft Computing*, 11(2):2472–2486, 2011. ISSN 1568-4946. (Cited on page 150.)

[283] V. Ravi, H. Kurniawan, Peter Nwee Kok Thai, and P. Ravi Kumar. Soft computing system for bank performance prediction. *Applied Soft Computing*, 8(1): 305–315, 2008. ISSN 1568-4946. (Cited on page 150.)

[284] Chih-Fong Tsai and Ming-Lun Chen. Credit rating by hybrid machine learning techniques. *Applied Soft Computing*, 10(2):374–380, 2010. ISSN 1568-4946. (Cited on page 150.)

[285] Javier De Andrés, Pedro Lorca, Francisco Javier de Cos Juez, and Fernando Sánchez-Lasheras. Bankruptcy forecasting: A hybrid approach using Fuzzy c-means clustering and Multivariate Adaptive Regression Splines (MARS). *Expert Systems with Applications*, 38(3):1866–1875, 2011. ISSN 0957-4174. (Cited on page 150.)

[286] António Afonso, Pedro Gomes, and Philipp Rother. Short- and long-run determinants of sovereign debt credit ratings. *International Journal of Finance & Economics*, 16(1):1–15, 2011. ISSN 1099-1158. (Cited on page 152.)

[287] International Monetary Fund. Global Financial Stability Report: Sovereigns, Funding, and Systemic Liquidity. World economic and financial surveys, Washington DC, U.S.A., 2010. (Cited on page 152.)

[288] N. Landwehr, M. Hall, and E. Frank. Logistic model trees. *Machine Learning*, 59 (1-2):161–205, 2005. (Cited on pages 155, 167, and 168.)

[289] David Hauner, Jiri Jonas, and Manmohan Singh Kumar. Sovereign Risk: Are the EU's New Member States Different? *Oxford Bulletin of Economics and Statistics*, 72(4):411–427, 2010. ISSN 1468-0084. (Cited on page 157.)

[290] Bernhard Rauch, Max Gottsche, Gernot Brahler, and Stefan Engel. Fact and fiction in EU-governmental economic data. *German Economic Review*, 12(3):243–255, 2011. (Cited on page 158.)

[291] P.A. Gutierrez, S. Salcedo-Sanz, C. Hervás-Martínez, L. Carro-Calvo, J. Sánchez-Monedero, and L. Prieto. Evaluating nominal and ordinal classifiers for wind speed prediction from synoptic pressure patterns. In *Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on*, pages 1265–1270, 2011. (Cited on page 161.)

[292] John K. Kaldellis and D. Zafirakis. The wind energy (r)evolution: A short review of a long history. *Renewable Energy*, 36(7):1887 – 1901, 2011. ISSN 0960-1481. (Cited on page 162.)

[293] R. Saidur, M.R. Islam, N.A. Rahim, and K.H. Solangi. A review on global wind energy policy. *Renewable and Sustainable Energy Reviews*, 14(7):1744–1762, 2010. ISSN 1364-0321. (Cited on page 162.)

[294] M. Khashei, M. Bijari, and G.A. Raissi Ardali. Improvement of auto-regressive integrated moving average models using fuzzy logic and artificial neural networks (anns). *Neurocomputing*, 72(4-6):956–967, 2009. (Cited on page 162.)

[295] J.L. Torres, A. García, M. De Blas, and A. De Francisco. Forecast of hourly average wind speed with ARMA models in Navarre (Spain). *Solar Energy*, 79 (1):65–77, 2005. (Cited on page 162.)

[296] T.G. Barbounis and J.B. Theocharis. Locally recurrent neural networks for long-term wind speed and power prediction. *Neurocomputing*, 69(4-6):466–496, 2006. (Cited on page 162.)

[297] A. Costa, A. Crespo, J. Navarro, G. Lizcano, H. Madsen, and E. Feitosa. A review on the young history of the wind power short-term prediction. *Renewable and Sustainable Energy Reviews*, 12(6):1725–1744, 2008. (Cited on page 162.)

[298] M.A. Mohandes, T.O. Halawani, S. Rehman, and A.A. Hussain. Support vector machines for wind speed prediction. *Renewable Energy*, 29(6):939–947, 2004. (Cited on page 162.)

[299] G. Li, J. Shi, and J. Zhou. Bayesian adaptive combination of short-term wind speed forecasts from neural network models. *Renewable Energy*, 36(1):352–359, 2011. (Cited on page 162.)

[300] Z.H. Chen, S.Y. Cheng, J.B. Li, X.R. Guo, W.H. Wang, and D.S. Chen. Relationship between atmospheric pollution processes and synoptic pressure patterns in northern china. *Atmospheric Environment*, 42(24):6078–6087, 2008. (Cited on page 162.)

[301] W.-L. Cheng. Synoptic weather patterns and their relationship to high ozone concentrations in the Taichung Basin. *Atmospheric Environment*, 35(29):4971–4994, 2001.

[302] S. Osowski and K. Garanty. Forecasting of the daily meteorological pollution using wavelets and support vector machine. *Engineering Applications of Artificial Intelligence*, 20(6):745–755, 2007.

[303] A. Paniagua-Tineo, S. Salcedo-Sanz, C. Casanova-Mateo, E.G. Ortiz-García, M.A. Cony, and E. Hernández-Martín. Prediction of daily maximum temperature using a support vector regression algorithm. *Renewable Energy*, 36(11):3054–3060, 2011.

[304] D. Paredes, R.M. Trigo, R. Garcia-Herrera, and I.F. Trigo. Understanding precipitation changes in Iberia in early spring: Weather typing and storm-tracking approaches. *Journal of Hydrometeorology*, 7(1):101–113, 2006.

[305] R. Romero, G. Sumner, C. Ramis, and A. Genovés. A classification of the atmospheric circulation patterns producing significant daily rainfall in the Spanish Mediterranean area. *International Journal of Climatology*, 19(7):765–785, 1999.

[306] R.M. Trigo and C.C. DaCamara. Circulation weather types and their influence on the precipitation regime in Portugal. *International Journal of Climatology*, 20 (13):1559–1581, 2000. (Cited on page 162.)

[307] L. Carro-Calvo, S. Salcedo-Sanz, N. Kirchner-Bossi, A. Portilla-Figueras, L. Prieto, R. Garcia-Herrera, and E. Hernández-Martín. Extraction of synoptic pressure patterns for long-term wind speed estimation in wind farms using evolutionary computing. *Energy*, 36(3):1571–1581, 2011. (Cited on page 162.)

[308] L. Carro-Calvo, S. Salcedo-Sanz, L. Prieto, N. Kirchner-Bossi, A. Portilla-Figueras, and S. Jiménez-Fernández. Wind speed reconstruction from synoptic pressure patterns using an evolutionary algorithm. *Applied Energy*, 89(1):347–354, 2012.

[309] M. Burlando, M. Antonelli, and C.F. Ratto. Mesoscale wind climate analysis: Identification of anemological regions and wind regimes. *International Journal of Climatology*, 28(5):629–641, 2008.

[310] F.O. Hocaoğlu, T.N. Gerek, and M. Kurban. A novel wind speed modeling approach using atmospheric pressure observations and hidden Markov models. *Journal of Wind Engineering and Industrial Aerodynamics*, 98(8-9):472–481, 2010. (Cited on pages 162, 168, and 169.)

[311] C. Soriano, A. Fernández, and J. Martin-Vide. Objective synoptic classification combined with high resolution meteorological models for wind mesoscale studies. *Meteorology and Atmospheric Physics*, 91(1-4):165–181, 2006. (Cited on page 162.)

[312] E. Kalnay, M. Kanamitsu, R. Kistler, W. Collins, D. Deaven, L. Gandin, M. Iredell, S. Saha, G. White, J. Woollen, Y. Zhu, M. Chelliah, W. Ebisuzaki, W. Higgins, J. Janowiak, K.C. Mo, C. Ropelewski, J. Wang, A. Leetmaa, R. Reynolds, R. Jenne, and D. Joseph. The NCEP/NCAR 40-year reanalysis project. *Bulletin of the American Meteorological Society*, 77(3):437–471, 1996. (Cited on page 164.)

[313] Earth System Research Laboratory. Physical Sciences Division. U.S. Department of Commerce-National Oceanic and Atmospheric Administration. The ncep/ncar reanalysis project, 2013. URL http://www.esrl.noaa.gov/psd/data/reanalysis/reanalysis.shtml. (Cited on page 164.)

[314] I.T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer, 2nd edition, Oct 2002. (Cited on page 166.)

[315] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993. (Cited on page 167.)

[316] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1996. (Cited on page 167.)

[317] S. le Cessie and J.C. van Houwelingen. Ridge estimators in logistic regression. *Applied Statistics*, 41(1):191–201, 1992. (Cited on page 168.)

[318] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 38(2):337–374, 2000. (Cited on page 168.)