Utility-based Regression

Rita Paula Almeida Ribeiro

Supervisor: **Prof. Luís Torgo**

Department of Computer Science
Faculty of Sciences University of Porto
2011

Utility-based Regression

Rita Paula Almeida Ribeiro

Thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science

Department of Computer Science

Faculty of Sciences University of Porto

2011

Thesis Committee:

Prof. Miguel Filgueiras, University of Porto

Prof. Peter Flach, University of Bristol

Prof. Bernhard Pfahringer, University of Waikato

Prof. Paulo Azevedo, University of Minho

Prof. Alípio Jorge, University of Porto

Prof. Luís Torgo, University of Porto

"Every adversity has the seed of an equal or greater benefit."

Napoleon Hill (1883 - 1970)

Acknowledgments

This thesis is the result of a long hard work for which many people have contributed, directly and indirectly.

I start by thanking my supervisor Professor Luís Torgo for making possible the production of this thesis and for introducing me to the areas of Data Mining and KDD, which have interested me so much. I have learnt a lot with him, especially with his scientific knowledge.

I must also express my gratitude to my colleagues at LIAAD INESC PORTO L.A. and abroad researchers who have given me different insights of life and many enjoyable moments.

A brief acknowledge to "Piled Higher and Deeper" Comic Strips by Jorge Cham for providing me so many good laughs and for making me feel that "I'm not alone"!

A note of gratefulness to all, friends and gurus!, who have given me strength and support during this period.

A very special word of appreciation to my family for giving me the values that have guided me throughout all these years.

I end up expressing my thanks to Jorge: his love, affection and confidence meant everything.

The work presented in this thesis was partially funded by the Portuguese Science and Technology Foundation (FCT) with the PhD grant SFRH/BD/17111/2004 and by FEDER funds through COMPETE program and by FCT national funds in the context of the project MORWAQ (PTDC/EIA/68489/2006). Part of the results were obtained with the help of the high-performance computing cluster of the Center for Research in Advanced Computing Systems (CRACS) created with the support of FCT.

Rita P. Ribeiro

Porto, 2011

Abstract

In many real world data mining prediction tasks, such as the forecast of extremely profitable stock trading actions, fraud detection in credit card transaction, ecological or meteorological catastrophes, among others, the phenomenon to predict is described by a specific range of values.

Given the nature of this type of applications, the relevance of the values forecasted may lead to trigger alarms, preventive measures, among others, involving, inevitably some costs. When understanding the particularities of this type of applications, one can acknowledge that the target variable does not have a uniform relevance throughout its domain.

Utility-based mining is a relatively recent approach to learning problems that involve costs and/or benefits. It has emerged from the already known cost-sensitive learning, as the most close approach to the representation of real world problems. Most of the work conducted in this area is related to classification problems. However, we claim that in some of these applications the phenomena to predict is, in its essence, numeric. This means that the target variable is continuous and, therefore, the same type of problems can appear in regression.

There are many real world applications in the conditions mentioned above: the stock market financial prediction is one of such examples. The more relevant values will correspond to the most extreme and rare price alteration. These subsets of values will be the ones that will potentially give us the most helpful information.

In this thesis, we study the general problem of regression based on utility. More conventional regression techniques assume that the relevance of values throughout the objective variable is uniform and that the magnitude of the predictions error is the only factor related to cost. Along this study we show that an evaluation based on utilities is more suitable when dealing with regression tasks where the target variable does not have a uniform relevance. Even though the importance of phenomena is often related to its rarity, that might not always be the case. For this reason, our utility approach is based on any continuous relevance function defined to the target variable. We propose a new evaluation methodology that assesses the utility (cost / benefit) of a prediction for a given value of the target variable, based on the prediction error and on the relevance of both predicted and true values.

When the goal is to predict a rare value, there are typically crucial decisions associated to those predictions (e.g. trading actions, trigger different type of alarms). In such context, it is important

to evaluate the models for those values that really matter, i.e. those that describe the target rare event. Moreover, it may be important to evaluate the models from a ranking perspective towards the target event. For this reason, we also derive utility-based evaluation metrics better designed to cope with rarity (e.g. precision, recall, Precision-Recall curves). We illustrate the advantage of using such metrics in the context of two-real world applications.

Finally, we propose ubaRules, a regression rules ensemble system that incorporates a measure based on utility (derived from the proposed methodology) as a preference criterion in the creation of models that satisfy the applications' requirements: we intend to derive precise and interpretable regression models.

Resumo

Existem muitos problemas de previsão reais em *data mining* onde o fenómeno a prever é descrito por um intervalo específico de valores. A previsão de operações no mercado bolsista que originem lucros avultados, a detecção de fraudes nas transacções de cartões de crédito, antevisão de catástrofes ecológicas ou meteorológicas, constituem alguns exemplos. Dada a natureza deste tipo de aplicações, a relevância dos valores previstos pode desencadear o accionamento de alarmes e / ou medidas preventivas o que, inevitavelmente, envolve custos. Uma das particularidades deste tipo de aplicações tem a ver com o facto da variável objectivo não possuir uma relevância uniforme ao longo de todo o seu domínio.

Utility-based mining é uma abordagem relativamente recente a problemas de aprendizagem envolvendo custos e/ou benefícios. Tendo por base a aprendizagem sensível ao custo, esta abordagem surge como uma evolução, fazendo uma representação mais real do problema. A maior parte do trabalho feito nesta área está relacionado com problemas de classificação. No entanto, em algumas destas aplicações, o fenómeno a prever é, na sua essência, numérico. Isto quer dizer que a variável objectivo é contínua e, portanto, o mesmo tipo de problemas pode surgir também no contexto da regressão. Existem muitos exemplos reais de aplicações nestas condições, tais como a previsão no mercado bolsista. Neste caso, os valores mais relevantes correspondem às variações de preços mais raras e também mais extremas. Estes subconjuntos de valores serão aqueles que potencialmente nos fornecerão a informação mais útil. É portanto sobre este conjunto de valores que interessa obter previsões precisas.

Nesta tese, estudamos o problema da regressão baseado na utilidade. Técnicas convencionais de regressão assumem que a relevância dos valores ao longo da variável objectivo é uniforme e que a magnitude do erro de previsão é o único factor associado ao custo. Ao longo deste estudo, mostramos que a avaliação baseada na utilidade é mais adequada quando se trata de tarefas de regressão, onde a variável objectivo não tem uma relevância uniforme. Ainda que a importância do fenómeno seja, tipicamente, associada á sua raridade, nem sempre isso acontece. Por esta razão, a nossa abordagem de utilidade é baseada numa qualquer função contínua de relevância definida para a variável objectivo. Propomos uma nova metodologia de avaliação, que afere a utilidade (custo/benefício) de uma previsão para um dado valor da variável objectivo, baseada no erro de previsão e na relevância dos valores previstos e dos valores verdadeiros.

No contexto da previsão de valores raros, existem, tipicamente, decisões importantes associadas a essas previsões (e.g. acções de compra e venda no mercado bolsista accionam diferentes tipos de

alarme). Neste sentido, é importante avaliar os modelos para os valores que realmente interessam, isto é, aqueles que descrevem o evento raro da variável objectivo. Para além disso, pode existir o interesse de avaliar os modelos numa perspectiva de *ranking* em relação ao evento objectivo, para uma selecção mais adequada dos modelos. Por esta razão, derivamos métricas de avaliação baseadas na utilidade que constituem uma adaptação das métricas mais adequadas à raridade (e.g. precision, recall, Precision-Recall curves). Ilustramos a vantagem de usar essas métricas no contexto de duas aplicações do mundo real.

Por fim, propomos um sistema de regras de regressão conjugadas que incorpora uma medida baseada na utilidade (derivad da metodologia proposta) como critério preferencial para modelos que satisfazem os requisitos da aplicação: tencionamos derivar modelos de regressão precisos e interpretáveis.

Contents

List of Tables

List of Figures

List of Examples

List of Algorithms

- 1 -

Introduction

"No sensible decision can be made any longer without
taking into account not only the world as it is,
but the world as it will be ..."
Isaac Asimov (1920 - 1992)

In our everyday life most of our actions are ruled by decisions we consider useful for us. Our objective is to make the best decisions so that we achieve the maximum utility at the end of a day, a week, a month and so on. If we think of a large-scale problem, the objective remains the same: choose the proper decisions to maximize the utility, but the risk we incur becomes higher. Considerable profits or benefits can be achieved, but also serious or costly consequences. For these more complex scenarios, models are built based on previous experience, being aware of costs/benefits involved on a decision. With these models we then make predictions about the "best" decision to take. Our final decision should be the one which is predicted to lead to the maximum utility value, among all possible predictions.

This thesis studies the problem of Utility-based Regression. Our target problem is to predict the value of a continuous variable based on its relationship with a set of variables, maximizing some utility metric.

In this first chapter, we introduce the problem in the field of Knowledge Discovery research. We also indicate our main motivations and contributions.

1.1 Knowledge Discovery and Data Mining

Data is everywhere in our life and storing it has always been a concern. One of the main reasons to do so was to eventually extract potential interesting information from it using it in an advantageous way. To acquire knowledge from data, there are two major types of learning techniques:

- *Supervised learning*: the objective is to learn a model from the given observations which describe the causal relationship between a target variable - called dependent variable - and a set of variables - called the independent variables; classification, regression and time series are three of the most well-known predictive tasks;
- *Unsupervised learning*: the objective is to learn a model which describes a set of patterns found in the given observations; clustering and association rules are two examples of such descriptive techniques.

Over the last decade, as the volume of the data collected by organizations on their activity grows, data mining became very popular. The data mining task comprehends the process of data analysis and the usage of learning techniques that yield to the discovery of knowledge in the data. The data mining process culminates on the generation of a model/theory which may describe a pattern or a causal relationship which should produce useful outputs when applied to new data, from the perspective of the end-user of the data mining application.

Data mining is currently used in many application areas. Banking, insurance, sales departments of companies and medicine, are some of the areas which commonly use it. Banks and insurance companies use data mining techniques as a decision support tool for small credit loans or insurance contracts approvals, thus aiming to reduce their costs. Through data mining is also possible to perform customer segmentation, a valuable information for any marketing department of a company to increase the sales. Medical research institutions frequently use data mining to develop their tests on medication effects.

Nevertheless, data mining tasks are not always about learning what is standard and common. Many applications have emerged in areas where what is really interesting for the end-user is to capture and to understand what is anomalous or, at least, surprising (e.g. finance, fraud, security, ecology, meteorology). In fact, reality is not perfect, quite on the contrary, it is full of exceptions and results from multiple external factors, which prevents it from being easily described by any mathematical model. These application areas bring new challenges to the standard learning techniques, namely regarding the standard evaluation measures used by them.

1.2 Context and Definition of The Problem

This thesis is focused on a special type of applications: data mining tasks where the main goal is the predictive performance on a subset of values of a continuous target variable. Many real-world applications from several scientific areas share this objective. This is the case of the prediction

of highly profitable trading actions, big credit card transactions, ecological/meteorological catastrophes, among others.

Given the nature of this type of applications, the relevance of the values within specific ranges of the continuous target variable often imply costly decisions. The anticipation of critical phenomena in such domains can, for instance, trigger preventive actions and thus, is considered of high importance.

Cost-sensitive learning (e.g. Domingos (1999); Elkan (2001); Zadrozny et al. (2006)) is a key technique for addressing many real world data mining applications, which have similar characteristics. However, most of the work done using these techniques, is focused on classification problems.

Here, we are specially interested in those applications where the phenomenon to predict is essentially numeric. This means that the target variable we aim at predicting is continuous and, thus, our focus is on regression problems.

The prediction of stock market returns is one of the examples we can find in finance. In this case, the target of prediction is a numerical value. Moreover, the more relevant values are the extreme ones which are also the most unusual ones. In fact, as it will be further shown, this is a quite common association: rarity and high relevance values. Nevertheless, it should be remarked that though it is a frequently observed association, rarity and relevance have no fixed relationship.

1.3 Motivation and Main Contributions

In this thesis, our main goal is to address a particular class of regression tasks. The peculiarity of these tasks is the existence of differentiated costs/benefits associated with the predictions made across the domain of the target variable.

Traditionally, it is assumed that in regression the determinant factor to express the quality of a prediction is the magnitude of the errors. One of the main claims of this thesis is that this assumption leads to sub-optimal results in our target applications.

The current state of the art on cost-sensitive learning, which more recently evolved to utility-based mining, is focused on classification tasks. However, the same type of cost-sensitive applications can appear in regression tasks. Providing a way of addressing them through concepts of utility-based mining is the main objective of this thesis.

Handling cost-sensitive applications has lead to two main approaches: a) developing new learning algorithms; b) re-sampling the available data in order to make the problem treatable by existing algorithms. In this thesis we have followed the first path, that we think is more in accordance with cost-sensitive principles. This type of approach requires the development of new evaluation metrics that are to be optimized by the new learning methods.

Our work has lead to the following five main contributions:

i. increase the consciousness of the data mining research community for these type of applications and in general to cost-sensitive regression;

ii. explain why the standard regression evaluation metrics are not effective in this context;

iii. propose a new evaluation methodology based on the concept of utility that addresses the prediction of a continuous variable with non-uniform costs/benefits along its domain;

iv. based on the utility formulation, derive concepts related to the analysis of precision and recall in regression; moreover, show how we can extend ROC analysis or Precision-Recall Curves to regression;

v. finally, present a new learning system which uses the previously proposed evaluation metrics as preference criteria to build models biased towards the objectives of our target applications; the system was designed to produce accurate and interpretable models.

1.4 Organization of the Thesis

The thesis is structured in seven chapters outlined below.

Chapter 1 - Introduction
In the present chapter, we have described the problems addressed in this thesis in the context of knowledge discovery from data. Main motivations, challenges and contributions of our work were also outlined.

Chapter 2 - Utility-based Mining
In the next chapter, we give a small survey on utility-based mining. We analyze it from a broader perspective. We discuss aspects such as cost-sensitive learning, outlier mining and highly skew class distribution problems. These are all problems related to the proposals presented in the remaining of the thesis.

Chapter 3 - Utility-based Regression

In Chapter 3, we introduce our proposal of utility-based regression. We motivate the need for this new evaluation methodology as a fulfillment of some real-world applications requirements. At the end of the chapter, we provide an application problem to better illustrate the advantages of our proposal in comparison to standard evaluation measures.

Chapter 4 - Prediction of Rare Values in Regression

In Chapter 4, we use some of the concepts defined on Chapter 3 to map the concepts of *precision* and *recall* into a regression context. These are key concepts to address prediction problems where the target is accuracy on rare events. From these concepts we show how to derive ROC and Precision-Recall graphs. Again, we end this chapter by giving a small illustrative example.

Chapter 5 - ubaRules: Utility-based Regression Rule Ensembles

In Chapter 5, we present ubaRules, an utility-based regression rules learner. The goal of this system is to give accurate and interpretable predictions for our target applications, by applying the evaluation measures presented in Chapters 3 and 4. Each of the techniques used in the construction of this system is described in detail.

Chapter 6 - Empirical Analysis of ubaRules

In Chapter 6, we describe the experimental evaluation of ubaRules on several problems.

Chapter 7 - Conclusions

Chapter 7 concludes the thesis. We summarize what motivated us and the contributions achieved. We end up by giving some further research directions.

1.5 Bibliographic Note

Some of the work described in this thesis has been published elsewhere. The following list gives a short reference to these publications.

- Chapter 3:
  - Ribeiro, R. P. and Torgo, L. (2008b). Utility-based performance measures for regression. In et al, K. W., editor, *The 3rd Workshop on Evaluation Methods for Machine Learning.* held in the context of the 25th International Conference of Machine Learning (ICML ' 08)

  - Torgo, L. and Ribeiro, R. P. (2007). Utility-based regression. In et al, K. J. N., editor, *PKDD'07: Proceedings of 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 4701 of *LNCS*, pages 597–604. Springer

- Chapter 4:

  - Torgo, L. and Ribeiro, R. P. (2009). Precision and recall in regression. In Gama, J., Costa, V. S., Jorge, A. M., and Brazdil, P., editors, *DS'09: 12th International Conference on Discovery Science*, volume 5808 of *LNCS*, pages 332–346. Springer
  - Ribeiro, R. P. and Torgo, L. (2008a). A comparative study on predicting algae blooms in douro river, portugal. *Ecological Modelling*, 212(1-2):86–91. Selected Papers from the 5th European Conference on Ecological Modelling. Elsevier
  - Ribeiro, R. P. and Torgo, L. (2007). An effective evaluation metric for forecasting microalgae blooms. Technical report, LIAAD INESC PORTO LA

- Chapter 5: and Chapter 6:

  - Ribeiro, R. P. and Torgo, L. (2006). Rule-based prediction of rare extreme values. In et al, T. L., editor, *DS'06: Proceeding of the 9th International Conference on Discovery Science*, volume 4265 of *LNCS*, pages 219–230. Springer. Carl Smith Best Student Paper Award sponsored by Yahoo! Research

- 2 -

Utility-based Mining

Utility
"that property in any object, whereby it tends
to produce benefit, advantage, pleasure, good, or happiness...or
...to prevent the happening of mischief, pain, evil, or unhappiness"
Jeremy Bentham (1748 - 1832)

The term *utility* was first used in complexity economics to represent the benefits arising from the consumption of goods or services in a economy. Recent studies within data mining have found this to be also applicable to the data mining process. In all stages of this process (e.g. data acquisition, data modelling, model application), there is an utility factor involved, which will end by having an impact in the final utility of the results. The main principle of Utility-based Data Mining is to maximize the expected utility associated to the entire data mining process by improving the utility results in one or more of these stages.

In this chapter, we give a brief overview of the research carried out around Utility-based Data Mining, focusing particularly on the data modelling stage. We present some main applications and techniques of this area. We also give some insights on how the concept of utility is extendable to many current fields of research.

2.1 Introduction

The study of many real-world applications by the data mining community, and the recent interest in tasks involving the prediction of important phenomena (e.g. ecological/meteorological catastrophes, frauds, network intrusions, stock market forecast), made clear that real-life is, in fact, full of non-uniform costs.

The importance assigned to a phenomenon can arise from any application-specific feature. It can be due to its profitability, rarity or even for its catastrophic consequences. Whatever is the case, it

is essential to notice that decisions based upon standard models' predictions may no longer be the optimal ones. The predictions should lead to decisions that optimize a domain-appropriate metric. Thus, selecting the most appropriate performance measures is application dependent.

Utility Based Data Mining (UBDM), as it is presented by Weiss et al. (2005, 2008); Zadrozny et al. (2006), addresses this challenge by considering utility as a new general metric to maximize. The utility should be defined according to the application problem specifications. The concept - originally employed in economics - comprehends both benefits and costs (negative benefits).

Utility-based research covers a set of wide techniques in data mining, namely, cost sensitive learning, detection of rare events (e.g. catastrophes), prediction of rare events, i.e. events which are rare but for which there is some supervised information about them (e.g. credit card or telecommunication fraud), the detection of new type of events through anomaly detection (e.g new network intrusions), association rules to discover new relationships, and so on.

Depending on the application, outliers can be specially meaningful values and its detection may be considered of high utility. Our work focus on this type of applications in which outliers, or some other special subset of values, may be of major importance and, therefore, in this context, so the cost-sensitive techniques assume equal relevance. Our objective is to provide a solution for regression domains based on related work existing for classification domains. For this reason, on this chapter, we will start by explaining how cost-sensitive learning has evolved to utility-based learning. Afterwards, we present a brief survey on methods of outlier detection. Due to space limitations, and as these areas are still an active line of research, we are confining to those techniques which are related at some point to our work.

2.2 Background in Cost-Sensitive Learning

In data mining we can identify several types of costs: misclassification costs, data acquisition costs, computation costs and more. In many real world applications, the misclassification costs are the most important ones. This is the case of medical diagnoses, drug prescription to a patient, business management decision (e.g. retain *versus* production), targeted marketing, credit approval, prediction of highly profitable trading actions in stock markets, detection of fraud or intrusion in a network, or even the prediction of ecological or meteorological catastrophes, among others. For this type of applications, the goal of the prediction task is to minimize the total cost, noting that not all the misclassified examples have the same cost, as assumed for most standard classification algorithms.

For instance, in credit card fraud detection, missing a fraud is probably more expensive then issuing a false alarm. Moreover, in this particular application, the fraudulent cases are less then the non-fraudulent ones. For these reasons, it will not be wise to evaluate models for this task under a cost-insensitive assumption, which associates the same cost to all the cases. For this concrete case, as for many other real-world applications, the class distributions of data are highly imbalanced - a fraud is, hopefully, a rare case. Cost-sensitive learning by handling of classification errors differently is also a common approach to solve the class imbalanced problem.

Regarding classification, and according to Zadrozny (2003) and Ling and Sheng (2010), there are three main approaches to handle non-uniform costs:

- **cost-sensitive learning by expected cost minimization**: associates costs to the classification errors and minimize the expected cost through the classifier learning method;
- **cost-sensitive learning by example weighting**: modifies the example distribution of the data set before applying the classifier learning method;
- **cost-sensitive classifiers**: adapts the standard classifier learning methods to make them cost-sensitive.

We briefly describe these approaches next.

2.2.1 Cost-sensitive Learning by Expected Cost Minimization

In a classification prediction task, we are given a set of observations $\mathcal{DS}$, which we call training set, of the form $\langle \mathrm{x}, y \rangle$ where x is a feature vector composed by the predictor (independent) variables and y is a class label of the target (dependent) variable. The target is known to be described by $f : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X}$ is the feature space and $\mathcal{Y}$ is the class label space. The goal of a classifier is to learn the best approximation to f, by optimizing some preference criterion and obtaining the model $\hat{f}_\Omega$ with a set of parameters designated by Ω.

In "standard" classification the objective is the minimization of the expected error rate.

Definition 2.1 (**Error Rate**)**.** The expected error rate of a classifier $\hat{f}_\Omega$ with respect to a set of instances $\langle \mathrm{x}, y \rangle \in \mathcal{X} \times \mathcal{Y}$ drawn from a distribution D is defined as

$$E_{ER}(\hat{f}_\Omega, D) \ := \ E_{\langle \mathrm{x}, y \rangle \sim D} \ [I(\hat{f}_\Omega(\mathrm{x}) \neq y)] \tag{2.1}$$

where I is an indicator function that returns the value 1 in case its argument is true and 0 otherwise.

Established this preference criterion, the optimal prediction $\hat{y} \in \mathcal{Y}$ for a given example $\mathrm{x} \in \mathcal{X}$ is determined by the classifier according to Equation 2.2.

$$\hat{y} \;=\; \underset{y \,\in\, \mathcal{Y}}{\operatorname{argmin}} \, E_{ER}(\hat{f}_{\Omega}, \langle \mathrm{x}, y \rangle) \tag{2.2}$$

This traditional formulation assumes that all the errors have the same cost, which as we have mentioned is not the case for many real-world applications. To address such situations the cost-sensitive meta-learning methods are the most commonly used algorithms. These methods are wrappers that convert a cost-insensitive learning technique into a cost-sensitive one, without modifying it. Through the cost matrix formulation, described by Domingos (1999) and Elkan (2001), one can incorporate the domain knowledge by associating costs to all possible classification scenarios.

Definition 2.2 (**Cost Matrix**)**.** Let $C := [c_{ij}]$ be a $n \times n$ cost matrix, where n is the number of classes existing in $\mathcal{Y}$. The value c_{ij} represents the cost of classifying an instance of true class j as class i. The structure of the cost matrix is such that,

$$c_{ij} = \begin{cases} 0, & \text{if } i = j; \\ > c_{ii}, & \text{otherwise.} \end{cases} \tag{2.3}$$

The entries in the cost matrix for the accurate predictions should have no cost (i.e. be equal to zero). The remaining entries express the costs associated to misclassifications and, thus should be a positive value.

Typically, costs are associated with decision-making problems (e.g. is the transaction a fraud?; is the patient sick?). According to the answer (e.g. yes or no) actions will be triggered. If we consider a cost-sensitive binary decision-making problem, the costs can be specified through a cost matrix as the one shown on Table 2.1. In this case, each instance of the problem is mapped to only one element of the set {yes, no}, - $\{1, 0\}$ in a binary notation - meaning that it is classified as positive or negative.

Table 2.1: A cost matrix for a binary classification problem.

2-class Cost Matrix			
		True	
		Negative	Positive
Predicted	Negative	c_{00}	c_{01}
	Positive	c_{10}	c_{11}

Whatever is the number of classes, within this costs setting, the objective is to minimize the expected cost of the classifier.

Definition 2.3 (**Classification Cost**)**.** The expected classification cost of a classifier $\hat{f}_\Omega$ with respect to a set of instances $\langle \mathrm{x}, y \rangle \in \mathcal{X} \times \mathcal{Y}$, drawn from a distribution D, is defined as

$$E_C(\hat{f}_\Omega, D) \;:=\; E_{\langle \mathrm{x},y\rangle \sim D}\,[c_{\hat{f}_\Omega(\mathrm{x}),y}] \tag{2.4}$$

where $C = [c_{ij}]$ is the cost matrix.

MetaCost (Domingos, 1999) is a largely known cost-sensitive meta learning algorithm that uses the *Bayes risk theory*. Assuming that we have the probability of each class y, for each example x, that is $P(y|\mathrm{x})$, then the *Bayes* optimal prediction for x is $\hat{y}$, the class with the lower expected cost (cf. Equation 2.6), i.e. with the minimum *conditional risk* R (cf. Equation 2.5).

$$R(x, y') \;=\; \sum_{y \,\in\, \mathcal{Y}} P(y|x) \cdot c_{y',y} \tag{2.5}$$

$$\hat{y} \;=\; \operatorname*{argmin}_{y \,\in\, \mathcal{Y}} R(x, y) \tag{2.6}$$

This algorithm is included in a set of algorithms which use *Bayes risk theory* and require the classifier to output class membership probabilities to assign each example to the class with the lowest expected cost. Nevertheless, this type of algorithm implies that the costs assigned to the classification errors and the class distribution are previously known. In fact, this is not usually the case, and in those situations both costs and class membership probabilities must be estimated and thresholds need to be established over these estimates in order to predict the class labels of the examples. In this sense, the accuracy of the classifier is very dependent on the "quality" of the estimates for the class membership probabilities.

According to researchers, though cost is the concept more traditionally used in cost-sensitive learning, the impact of a prediction should not be restricted to it. Besides implying no cost, accurate predictions may result in possible profit / benefit and thus should not be neglected. Benefits are considered a more natural baseline from where to measure all the benefits, whether positive or negative (commonly seen as costs). Benefits are, in fact, utility values expressed in some unit.

Elkan (2001) proposed a benefit matrix formulation where the costs are measured against a fixed baseline of benefit as it appears more natural for some domains. A benefit matrix obeys to the following formulation.

Definition 2.4 (**Benefit Matrix**). Let $B := [b_{ij}]$ be a $n \times m$ benefit matrix, where n is the number of classes existing in $\mathcal{Y}$. The value b_{ij} represents the benefit of classifying an instance of true class j as class i. The structure of the benefit matrix is such that

$$b_{ij} = \begin{cases} \geq 0, & \text{if } i = j; \\ < 0, & \text{otherwise.} \end{cases} \tag{2.7}$$

The benefit matrix is an extension to the cost matrix, in the way that its entries can have either positive or negative values. Furthermore, the diagonal elements of the matrix, representing the accurate predictions, should have non-negative values. The remaining elements of the matrix, representing opportunity costs (i.e. loss benefits) or false alarms, should have negative values.

In this context, a new objective was established in cost-sensitive learning, that is, maximizing the expected benefit. (cf. Equation 2.8).

Definition 2.5 (**Expected Benefit**). The expected classification benefit of a classifier $\hat{f}_\Omega$ with respect to a set of instances $\langle \mathrm{x}, y \rangle \in \mathcal{X} \times \mathcal{Y}$ drawn from a distribution D is defined as

$$E_B(\hat{f}_\Omega, D) \ := \ E_{\langle \mathrm{x},y \rangle \sim D} \ [b_{\hat{f}_\Omega(\mathrm{x}),y}] \tag{2.8}$$

where $B = [b_{ij}]$ is the benefit matrix.

The approaches referred until now assume that the costs are fixed, i.e. they only depend on the predicted and actual classes. Nonetheless, besides being difficult to determine the exact misclassification costs, studies in cost-sensitive learning (e.g. Elkan, 2001; Zadrozny and Elkan, 2001), argued that in real-world scenarios, costs should be example dependent. Consider the examples of credit card fraud detection and one-to-one marketing for donations. The amounts involved in both applications will affect significantly the costs in the first domain or the benefits of the second case.

To better illustrate this new formulation, Elkan (2001) described a simple example of a benefit matrix which we reproduce here in the Example 2.1.

Example 2.1. A benefit matrix: credit card transactions

Consider the credit card transactions domain. Table 2.2 expresses the benefit of a transaction as a function of the true type of the transaction, the predicted type of transaction by the bank and,

finally, the amount of the transaction (x) itself. The rationality behind this has to do with the costs/benefits for the bank after its prediction. The best scenario occurs when the bank approves a legitimate transaction, obtaining a profit of 2% of it. On the other hand, if it approves a fraudulent one, it looses all the money involved on it. The last two scenarios have non-trivial costs: if it refuses a legitimate transaction, it annoys the customer; if it refuses a fraudulent one it benefits, since it prevents a fraud. With this example, it is simple to realize that the cost or the benefit may not be constant, and may be distinct for different examples. In effect, according to Zadrozny and Elkan (2001), this example-dependent benefit property is what makes possible to achieve a good decision making. There is no constant c that maximizes the benefit for all x.

Table 2.2: A benefit matrix for credit card transactions.

Credit Card Fraud			
		Action	
		refuse	approve
Expected	fraudulent	\$20	-x
	legitimate	-\$20	$0.02x$

If the learning technique fails to consider the negative benefits (costs), it will lead to models performing poorly. At the same time, if the models are too conservative to avoid approving fraudulent transactions, then there is a high risk of generating models that are almost useless as they will obtain very poor benefits. In this sense, it is said that the models have large opportunity costs.

In this scenario, another problem formulation is proposed in cost-sensitive learning, which is to maximize the example-dependent benefits (cf. Equation 2.9).

Definition 2.6 (**Expected Example-dependent Benefit**)**.** The expected classification example-based benefit of a classifier $\hat{f}_\Omega$ with respect to a set of instances $\langle \mathrm{x}, y\rangle \in \mathcal{X} \times \mathcal{Y}$ drawn from a distribution D is defined as

$$E_{Bx}(\hat{f}_\Omega, D) \;:=\; E_{\langle \mathrm{x},y\rangle \sim D}\,[B(\mathrm{x}, bx_{\hat{f}_\Omega(\mathrm{x}),y})] \tag{2.9}$$

where $B : \mathcal{X} \times \mathbb{R} \to \mathbb{R}$ is the function that calculates the benefit of a prediction with respect to an example x and a benefit matrix $Bx := [bx_{ij}]$.

Following this criterion, the optimal prediction $\hat{y} \in \mathcal{Y}$ for an example $\mathrm{x} \in \mathcal{X}$ is obtained as presented in Equation 2.10.

$$\hat{y} = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} E_{Bx}(\hat{f}_\Omega, \langle \mathrm{x}, y \rangle) \tag{2.10}$$

2.2.2 Cost-sensitive Learning by Example Weighting

There are several ways of introducing *utility* in machine learning and doing it through the assignment of costs/benefits to classification errors is just one of them as referred by Abe (2005). In effect, all the approaches we have referred in the previous section, are applicable only if all the costs/benefits are known or possible to estimate. But there are many real-life applications where such task is not trivial.

In the approach described in this section, the conversion of classifier learning into a cost-sensitive algorithms is based on cost-proportionate weighting of the training examples, which can be realized either by feeding the weights to the classification algorithm, as often done in boosting (Freund and Schapire, 1997), or by careful sub-sampling. It is a more general approach as it does not require accurate probability estimates and can be combined with any existing learners.

For 2-class problems, Zadrozny et al. (2003) proposed the importance weighted classification, where each example is given a non-negative weight c that specifies the extra cost acquired in case of being misclassified. Instead of minimizing the error rate weighing all misclassifications under the same cost assumption, the objective is now to minimize the expected cost of the weighted classification.

Definition 2.7 (**Weighted Classification Cost**)**.** The expected weighted classification cost of a classifier $\hat{f}_\Omega$ with respect to a set of instances $(\mathrm{x}, y, c) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{C}$, where $\mathcal{C} \subset [0, \infty[$ is the importance or extra cost associated with a wrong classification to each instance, drawn from a distribution D is defined as

$$E_{WC}(\hat{f}_\Omega, D) := E_{(\mathrm{x},y,c)\sim D}\left[cI(\hat{f}_\Omega(\mathrm{x}) \neq y)\right] \tag{2.11}$$

where I is an indicator function.

This formulation of cost-sensitive learning in terms of one number per example is more general than cost matrix formulations which are more typical in cost-sensitive learning, when the output space is binary. On those situations, the only considered costs are true positives, false positives, true negatives and false negatives, i.e. the four possible outcomes of the cost matrix. This setting is more general in the sense that the importance may vary on a example-by-example basis. For a benefit setting, we can define the importance as the difference between classifying the example correctly and incorrectly. In that context, the objective changes to maximize the expected benefit and the same theory applies.

2.2.3 Cost-sensitive Classifiers

We can make classifiers cost-sensitive by changing their learning process so that they take in consideration the costs. There is a variety of learning methods where this technique has been applied.

The cost-sensitive decision trees proposed by Ling et al. (2004) use the misclassification costs in the split criterion. Some research has also been conducted over other learning techniques such as neural networks (e.g. Liu, 2006) and support vector machines (Yuanhong et al., 2009, e.g.), in order to make them cost-sensitive. Later, in this chapter, we will describe more cost-sensitive classifiers in detail.

2.3 Detection of Rare Cases

In several applications, rare events are often the more interesting ones from the knowledge discovery perspective. The occurrence of such events is registered by a small set of observations called outliers, which are significantly different from the rest of the observations in the data set.

Outliers are frequently seen as potential indicators of numerous critical situations such as fraudulent transactions, highly profitable stock market actions, ecological/meteorological catastrophes (e.g. flooding, natural fires, hurricanes), atypical disease symptoms and its possible spread, and identification of strange objects in surveillance systems. In this sense, the detection of outliers can lead to the discovery of unexpected, interesting and useful knowledge. From this perspective, outliers are points of high utility value.

2.3.1 Problem Definition and Challenges

The concept of outlier goes back to the origins of data analysis. The primary definition of outlier was given by Hawkins (1980),

Definition 2.8 (**Outlier - I**)**.** Outlier is "an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism".

Later, another definition of outlier was given by Barnett and Lewis (1994), which is also commonly used,

Definition 2.9 (**Outlier - II**)**.** Outlier is "an observation (or subset of observations) which appears to be inconsistent with the reminder of that set of data".

Initially, the identification of outliers had as only purpose data cleaning since they were considered errors. In this sense and not to wrongly influence the posterior statistical analysis, they were corrected or just expunged. Presently, researchers are also interested in understanding the "different mechanism" that is behind them. But, none of the above perspectives is wrong and it will always be hard to tell if a given outlier was originated unintentionally - representing an error, noise in the data - or intentionally - representing an unusual value originated from a natural variation of the population (cf. Figure 2.1). Either way, the identification of outliers is an important step in data analysis.

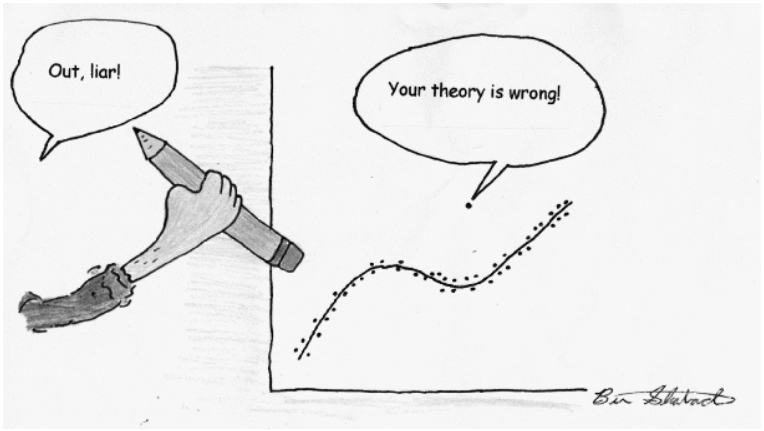


Figure 2.1: Is the Outlier a liar?

The importance of outlier detection is related to the fact they can represent critical information, which can trigger preventive or corrective actions in a large set of applications.

Outlier mining is an established data mining task and active line of research. It has two main objectives: defining the concept of outlier; and finding efficient methods to detect them. Even though it can appear as a common data mining task, there are several factors (Chandola et al., 2009) that make this task more challenging, namely:

- it is hard to define every possible "normal" behaviour;
- the boundary between normal and a outlying behaviour is often not precise;
- there is no such outlier general definition, as the exact notion of outlier varies according to the application domain;
- it is difficult to distinguish real meaningful outliers from simple random noise in data;

- in many applications, the outlier behaviour evolves with time;
- in many cases, outliers are the result of malicious actions, which try to make the outliers appear as normal situations and thus difficult their detection;
- there is an inherent lack of known labeled outliers, which adds more complexity to the outlier detection tasks.

Due to the above difficulties, the existing outlier detection techniques approach the problem by making assumptions on the type of target outliers, the kind of output required, and the nature of the data set.

Chandola et al. (2009) identified three types of outliers.

Definition 2.10 (**Point Outlier**)**.** Given a set of data instances, an instance is a point outlier if individually or in small groups is very different from the rest of the instances.

Credit card abnormal transactions of huge amounts are examples of point outliers.

Definition 2.11 (**Contextual Outlier**)**.** Given a set of data instances, an instance is a contextual outlier if when considered within a context is very different from the rest of the instances.

A temperature of 40° during winter time in Portugal is a contextual outlier, as it depends on the context where it is inserted, in this case: winter and Portugal.

Definition 2.12 (**Collective Outlier**)**.** Given a set of data instances, an instance is a collective outlier if, even though individually may not be an outlier, inspected in conjunction with related instances and with respect to the entire data set is an outlier.

The output of a human electrocardiogram is presented in some works (e.g. Chandola et al., 2009), as an example of collective outliers. The electrocardiograms have several flat lines, which can appear both at normal and outlier regions, the difference is that the later ones last longer. So, in order to detect abnormal behaviours it is necessary to analyse the entire electrocardiogram and thus, if there are outliers they are collective.

In what concerns the expected output from the outlier detection, two results can be returned with respect to a data point: a binary value stating if it is an outlier or not; a score, which indicates the degree of *outlyingness* of that point when compared to the rest of the points.

Regarding the number of attributes, outliers can be described by one attribute, in which case are called univariate, or described by a set of attributes, in which case are called multivariate, making its detection more complex.

Similarly to other learning tasks, outlier detection techniques can be divided in three main groups, as follows:

- *Supervised learning approaches*: require a data set with instances labeled as *normal* or *outlier* to then learn a 2-class model. Real life applications with all instances labelled are not easy to obtain and, moreover new outliers can always appear, different from the ones present in the data set.

- *Unsupervised learning approaches*: are able to identify outliers in non-labeled data set, using several techniques for that; these techniques are more general than the supervised approaches, as the non-labeled data is easier to obtain; for this reason it has been the area more extensively researched in the context of outlier detection.

- *Semi-supervised approaches*: in this technique we have both non-labeled and labelled instances of normal and outliers. The are several methods used to learn to detect outliers from this type of data. One of the used methods, requires that the normal instances are labeled during the modelling phase so that normality boundaries can be learned; every instance that does not fit the model is considered an outlier; while it is not required to get outlier labeled instances, getting a set of normal labeled instances is not easy for a real-life application.

The type of data set also determines the detection technique to use. High dimensional data sets pose computational efficiency issues. Complex data sets like sequence data, spatial data, streams, spatial-temporal and so on, require special techniques to be used.

In the following subsections, we briefly describe different outlier detection methods, for supervised and unsupervised learning methods, their application contexts and limitations. As we will see, each method defines its own outlier concept according to its detection technique, but they all express, essentially, the same concept presented in the Definitions 2.8 and 2.9.

2.3.2 Supervised Approaches

Supervised outlier detection techniques assume the existence of historical information on all the normal and outlier instances from where predictive models for outliers can be built. Most of the work regarding this area focus on classification tasks and, in particular, on binary classification as it considers only two classes: normal and outlier.

By the implicit definition of outlier, these classification tasks have an imbalanced class distribution, a well known problem and subject of research (e.g. Chawla, 2005; Kotsiantis et al., 2006; Kubat and Matwin, 1997). The instances with outlier class label are rare and, thus, far less frequent than the instances with the normal class label.

Learning algorithms that do not consider class-imbalance tend to be overwhelmed by the major class and ignore the minor one (Chawla, 2005). But, to the minority class, higher costs are typically assigned and, to be effective, the classifier must not neglect the performance on this minority class.

Within the set of applications with imbalanced class distribution, the subset of problems with a very high skew in the distribution is usually associated with domains with marked differences in terms of utility of the predictions. On these problems the rare classes usually describe the occurrence of rare phenomena. Examples of these problems include: prediction of fraudulent transactions (e.g. credit cards, mobile telecommunications), network intrusions, diagnose of rare diseases, ecological/meteorological catastrophes, and so on. Decisions aim at detecting rare but important cases. In these applications, the minority class is known as the positive class, and the majority class is the negative class. According to several studies (e.g. Weiss, 2004) this brings additional difficulties to traditional classification algorithms, due to:

- **lack of data**: the small number of examples of the minority class to learn from, limits the capacity of generalization of the standard learning method and causes the model to overfit the concept - this is known as the *small-disjuncts* [*1] problem;
- **the misclassification errors do not have all the same costs**: in traditional classifier learning it is assumed that all the misclassification errors are equal, but, typically, in real-life applications with class imbalance it is more serious to wrongly classify an example of the minority class than to misclassify an example of the majority class.
- **standard evaluation measures, such as accuracy, are no longer appropriate**: they minimize the overall error to which the minority class contribution is almost meaningless; the reason is that given the "natural" class imbalance distribution of the test set, the model will predict mostly the majority class and the minority class examples will be, with high probability, misclassified;

These are a few of the causes for the bad performance obtained by standard classification algorithms when predicting rare classes. Under these circumstances, there are two main approaches to tackle this type of problems:

1. find evaluation measures suitable for this type of classification problems that are able to measure and compare the performance of the classifiers according to the application objectives;

[*1] Details on *small-disjuncts* can be obtained in `http://storm.cis.fordham.edu/~gweiss/small_disjuncts.html`

2. make the classification algorithms aware of the application's objectives and bias their model conception according to it; this can be accomplished either by a wrapper approach, such as sampling the training data, or by changing the algorithm itself introducing a new preference criterion that can better guide the search process during model construction.

In the following subsections, we will briefly describe some of the solutions that have been proposed to the problem of class imbalance and, in particular, for the binary classification problem of rare class prediction.

2.3.2.1 Biased Evaluation Measures

One of the first things that is important to be considered refers to the inadequacy of the standard evaluation measures to the prediction of rare classes. An appropriate performance measure should be chosen.

The classifier maps instances into predicted classes. Thus, given a binary classifier and an instance, there are four possible outcomes that can be represented in a confusion matrix, such as the one shown in Table 2.3.

Table 2.3: A confusion matrix for a binary classification problem.

2-class Confusion Matrix				
		True		
		Negative	Positive	Total
Predicted	Negative	TN	FP	PNEG
	Positive	FN	TP	PPOS
	Total	NEG	POS	

The target class is the rare class and so is the positive class. The confusion matrix reports: the number of False Positives (FP) and True Positives (TP), i.e. the number of incorrect and correct predictions of the positive (rare) class; and the number of True Negatives (TN) and False Negatives (FN), i.e. the number of correct and incorrect predictions of the negative (common) class. This matrix contains information that forms the definition of many performance measures, enabling a quick inspection of the model performance.

Definition 2.13 (**Accuracy**)**.** Accuracy is the proportion of correct predictions made by the model. In a binary classification scenario it is defined by,

$$accuracy = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{2.12}$$

Even though *accuracy* is the standard performance measure most commonly used in classification, it shows to be ineffective regarding the class imbalance as it is illustrated in the following example.

Example 2.2. Inadequacy of *accuracy*: different diagnoses of a rare disease

Imagine that we have 100 patients to control the diagnose of a rare disease, and four different models to do it. The confusion matrices shown in Table 2.4 represent the predictions made by each of these models over the same group of patients. The rare disease corresponds to the minority/positive class.

Table 2.4: Four different classifiers represented by the confusion matrix of their predictions for the same data set. Each matrix gives an estimate of the accuracy of the models.

Model A Confusion Matrix			
		Disease	
		absent	present
Diagnose	negative	TN = 81	FN = 4
	positive	FP = 9	TP = 6

(a)

Model B Confusion Matrix			
		Disease	
		absent	present
Diagnose	negative	TN = 63	FN = 2
	positive	FP = 27	TP = 8

(b)

Model C Confusion Matrix			
		Disease	
		absent	present
Diagnose	negative	TN = 68	FN = 7
	positive	FP = 22	TP = 3

(c)

Model D Confusion Matrix			
		Disease	
		absent	present
Diagnose	negative	TN = 41	FN = 1
	positive	FP = 49	TP = 9

(d)

Models Performance				
	Model A	Model B	Model C	Model D
accuracy	0.87	0.71	0. 71	0.50

(e)

If we evaluate the performance of the four models, according to accuracy, we get to the values shown in Table 2.4e.

From the obtained results, model A is the best with 87% of accuracy*. Then, we have model B and C with the same* accuracy *value of 71%. This result is clearly counter intuitive given the goal of diagnosing effectively the rare disease. While model B correctly diagnosed 80% of the sick individuals, model C diagnosed only 30%. Model D achieved the worst* accuracy *result, only 50%. Still, this is the model that identifies more sick patients, both correctly and incorrectly.*

By assigning the same weight to all the cases, accuracy *diminishes the impact of the performance over the rare class. Therefore, the results shown in Table 2.4 are mainly driven by the TN, i.e.*

the correct diagnose of healthy individuals. Regarding this application, the cost of a false diagnose about the presence of the disease (FP) should have lighter consequences to the patients than a false diagnose about the absence of the disease (FN) as, this later one, can delay the start of a possible treatment.

Provost and Fawcett (1997) have shown that within the context of highly imbalanced class distribution, the average *accuracy* assumes that all errors, FP and FN, have the same impact, which is inappropriate. A suitable evaluation measure should focus on the *accuracy* over the minority class.

2.3.2.2 ROC Analysis

ROC analysis, and in particular its associated AUC measure, is the most commonly used evaluation framework to assess the overall quality of a classifier over imbalanced class distribution conditions. Provost et al. (1998) have proposed them as alternatives to *accuracy*.

The concept was initially presented by Egan (1975) in the context of the *Signal Detection Theory*. Its usefulness was later recognized in the field of medical diagnoses for its discriminatory power, and turned out to be a standard technique used in medicine and biology.

So far we have been referring to binary classifiers which assign a positive or a negative class to each example. Still, there is a set of classifiers called probabilistic classifiers (e.g. naive Bayes) which also address binary decision problems. A probabilistic classifier is a function $f : \mathcal{X} \rightarrow [0, 1]$ that maps each example x to a real number $f(\mathrm{x})$. A decision threshold t is then chosen to establish the positive examples, i.e. those examples for which $f(\mathrm{x}) \geq t$, and the negative examples, where all the remaining examples fit. Traditionally, the decision threshold is 0.5, but any other can be chosen. By changing the threshold, we get a new classifier. ROC analysis uses this technique to check the "quality" of a classifier under different conditions. It is also an important tool for evaluating and comparing classifiers when operating characteristics (i.e. class distribution and cost parameters) are imprecise, or not known, at training time.

ROC Curves

ROC curves establish, in the so-called ROC-space, the relation between the true positive rate and the false positive rate, which we define next.

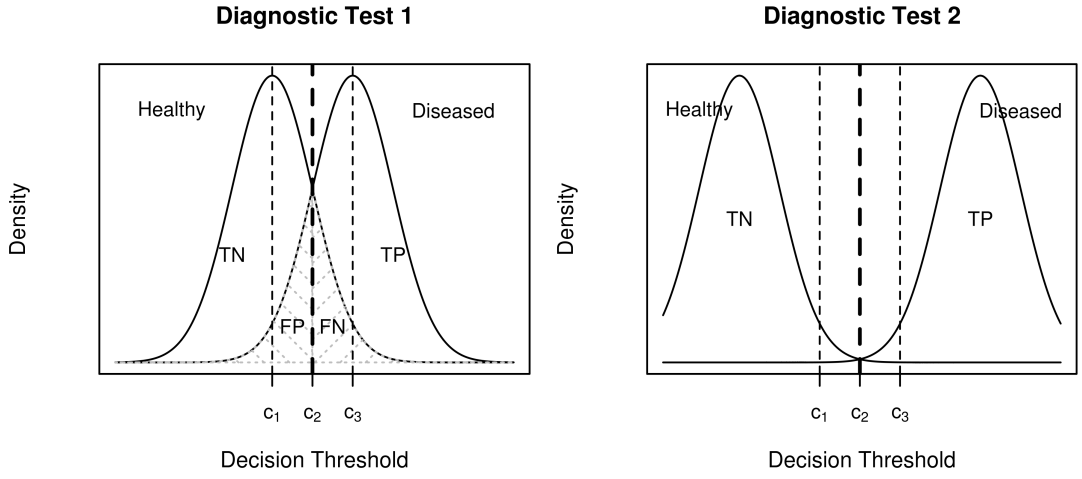
Definition 2.14 (**TPR**)**.** The True Positive Rate (TPR) (also designated as sensitivity, hit rate or recall) is the proportion of positive instances that were correctly captured by the model. It is defined by,

$$\text{TPR} = \frac{\text{TP}}{\text{POS}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.13}$$

Definition 2.15 (**FPR**)**.** The False Positive Rate (TPR) (also designated as specificity relative to the negative class) is the proportion of negative instances that were erroneously captured by the model as being positive. It is defined by,

$$\text{FPR} = \frac{\text{FP}}{\text{NEG}} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{2.14}$$

Through the ROC curves, one can visually inspect the ability of the classifier in distinguishing between false positives and true positives, independently of the class distribution of data and error costs. The reason is that each point in the ROC curve represents a different decision threshold, obtained from all the possible cutoffs along the scores, between positive and negative examples. According to the chosen threshold that assigns the class membership based on its estimates, so is obtained the distribution of TP, FP, TN, and FN. In Figure 2.2 we provide two hypothetical models (D_1 and D_2) for a medical diagnostic test with the respective density estimation of the distribution of the healthy and diseased patients. For each model, different ROC cut-offs values are considered. The scenario resulting from the cutoff c_2 for the model D_2, shown on the right side of the figure, corresponds to the ideal diagnostic test, where no errors are committed.



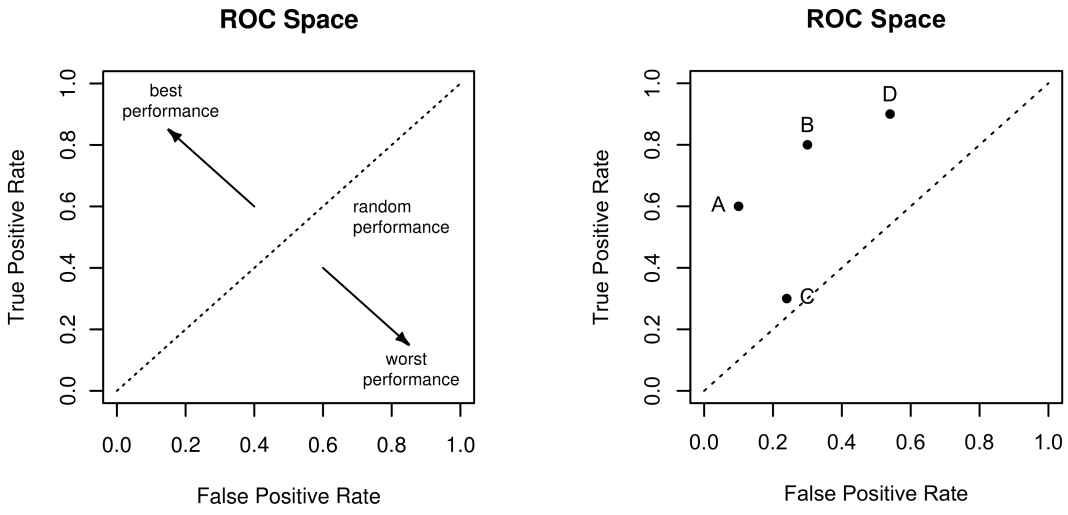
(a) c_2 has low probability of committing errors. (b) c_2 has almost no chance of committing no errors.

Figure 2.2: Choosing the best ROC cutoffs. Two possible distributions of healthy and diseased patients and the probable outcome for a medical diagnostic test on a new patient according to different ROC cutoffs.

ROC analysis presents, in fact, an advantage over the performance measures derived from the confusion matrix, such as *accuracy*, which are insensitive to class skew. In this sense, as ROC curves do not rely on a specific decision threshold, but rather try to represent all set of possible decision thresholds, the performance exhibited by the model ROC curve is not expected to change. ROC curves express the same information of a confusion matrix, but allow to generalize and visualize the performance of classifiers and compare them, whatever the class distribution or the misclassification costs are (e.g. Fawcett, 2004; Provost et al., 1998).

In this setting, an operating point is a specific combination of misclassification costs and class distributions. In particular, this technique meets the evaluation requirements of the prediction of rare classes - a particular case of the imbalanced class distribution problem.

In ROC space, illustrated in Figure 2.3 (a), models are characterized by the corner to which they are nearer. On the top left corner are the best models, the ones that classify all the positive examples correctly and do not have false positives. On the opposite corner, the bottom right corner, are the worst performance models, the ones that classify all the positive examples incorrectly. The bottom left corner corresponds to conservative models that classify all examples as negatives. On the top right are the "liberal" models, i.e. the ones that classify all examples as positive, but incur on many false positive errors, classifying all examples as positive, thus incurring on many false positive errors. Along the main diagonal are the random classifiers, the ones that perform no better than just random guessing.



(a) ROC Space Characterization

(b) Models Operating Points in the ROC space

Figure 2.3: ROC space characterization.

Through ROC analysis, we compared the models presented in Example 2.2 (page 21) and plotted them in the ROC space as shown in Figure 2.3 (b). According to the ROC space characterization,

one can first notice that models B and C have no longer equal performance. In fact, model C is a very poor performing model due to its low TPR value in comparison to model B. On the other hand, model D turned out to be one of the best classifiers, and this has to due with its high TPR. Model A with the best *accuracy* result shows also poor performance in the ROC space, due to its low TPR. Figure 2.3 (b) shows a single "operating point" per model. Usually, to draw ROC curves it is necessary to vary the decision threshold to generate several operating points of the classifiers.

Example 2.3. ROC curves: performance of classifiers of co-receptors used by HIV-1

The data set `ROCR.hiv` *(Sing et al., 2009) contains the predictions obtained by Support Vector Machines (*svm*) and Artificial Neural Network (*nnet*) to determine which of the co-receptors, CCR5 or CXCR4, HIV-1 used to infect the cells. This data set has an imbalanced class distribution as the positive class is associated to only 30% of the instances. Each classifier was subject to a 10-fold cross-validation technique (Stone, 1974). We used two variations of the two models, which we called* svm1 *and* nnet1*. The predictions of these two artificial models are the result of the manipulation of the original set of predictions obtained by each model. The goal was to obtain different, but still comparable models to the original ones. On* svm1*, we allocated the smaller errors of svm to the positive instances. On* nnet1*, we did the opposite and allocated the bigger errors of* nnet *to the positive instances.*

Each classifier shown in Figure 2.4 is the result of a vertical average (Fawcett, 2004) over all the ROC curves obtained by cross-validation. This means that the final curves for each classifier were obtained by averaging the TPR values at fixed FPR values.

The performance ranking of the classifiers can be captured visually. According to the ROC space characterization, from the top left corner to the bottom right corner, we enumerate the models consistently with their performance estimates from the best to the worst one. In this context, we have: svm1, with the best performance; the original svm and nnet; and, finally, with worst performance, nnet1. This ranking actually confirms our assumptions regarding the performance estimates of the artificial models, when we were generating them.

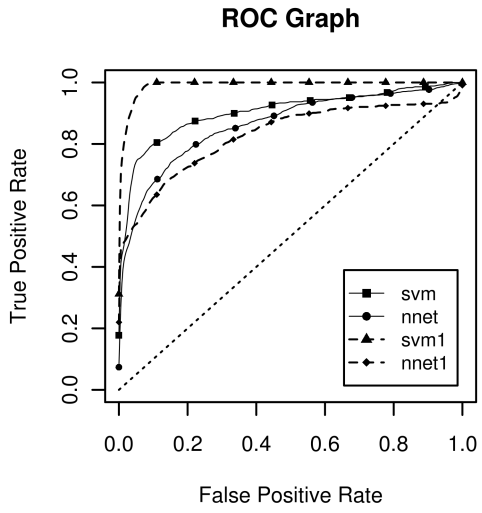


Figure 2.4: ROC curves of four different models. The relative positioning of the curves allows us to establish an empirical ranking among the models.

Choosing the best model using ROC curves can be very subjective. That is the case when there are too many models in comparison or when some curves mixed up together.

To obtain a less heuristic comparison of the performance of the classifiers based on ROC curves, two main techniques are used: AUC (Metz, 1978) and ROCCH (Provost and Fawcett, 1997, 2001).

Area Under the ROC Curve (AUC-ROC)

The Area Under the ROC curve (**AUC**) is a method used frequently when we want to represent the ROC curves performance information in a single-value.

AUC, as illustrated in Figure 2.5a, is a portion of the area of the unit square and, as so, its value is always be between zero and one, being defined by the following formula:

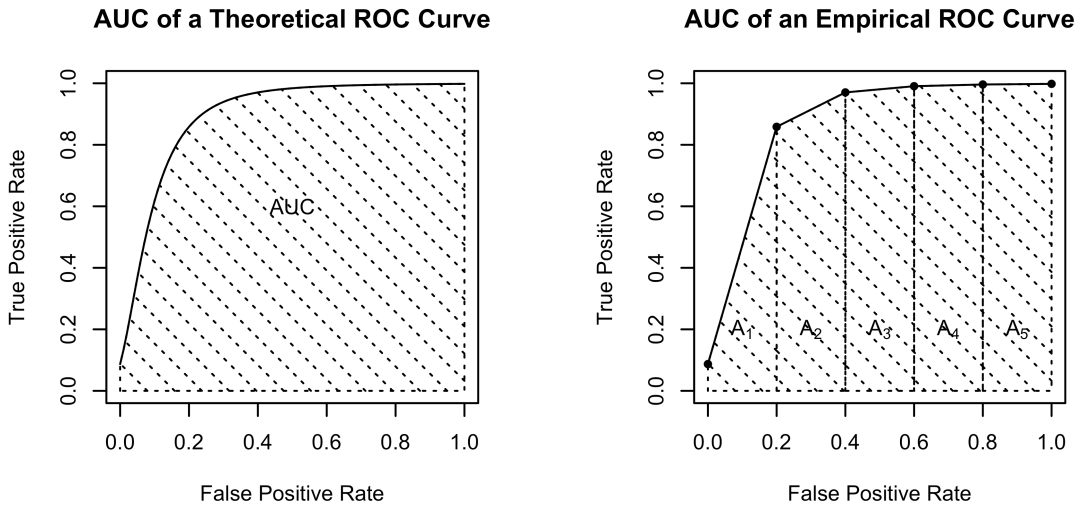
$$\text{AUC} = \int_0^1 \text{TPR}\, d\,\text{FPR} \tag{2.15}$$

To obtain the AUC value it is necessary to solve the above defined integral. There are some proposals (e.g. Krzanowski and Hand, 2009) based on specific distributional assumptions to solve it analytically. One of the most used techniques is the binormal model, which assumes a normal distribution for both the population of positive examples and the population of negative examples.

Nevertheless, if the ROC curve has been estimated empirically, its AUC is usually calculated with the trapezoidal rule (Davis and Goadrich, 2006). This rule approximates the area of a definite integral by splitting the interval of integration into sub-intervals and obtaining, for each of them, its trapezoidal area, as defined in Equation 2.16.

$$\int_a^b f(x)\,dx \;\approx\; \frac{b-a}{n}\left[\frac{f(a)+f(b)}{2}\sum_{k=1}^{n-1} f\left(a+k\frac{b-a}{n}\right)\right] \tag{2.16}$$

In terms of ROC analysis, the trapezoidal rule states that the AUC is the result of the sum of all the areas of the successive trapezoids between each ROC point of the empirical curve. In the concrete case of the ROC curve of Figure 2.5b, AUC is given by a trapezoidal rule with five sub-intervals defined by six ROC points. However, if there are many possible thresholds in a curve, this calculus may be quite an effort.



(a) (b)

Figure 2.5: The Area Under Curve (AUC) of an empirical ROC curve and a theoretical ROC curve.

For the case of empirical ROC curves, the numerical integration is unnecessary as AUC is equivalent to the probability that a randomly chosen positive example will have a smaller estimated probability of belonging to the negative class, than a randomly chosen negative example (cf. Equation 2.17). Hence, AUC can also be interpreted as the probability that the classifier represented by the curve will rank a randomly chosen positive instance higher than a randomly chosen negative instance. In this sense, AUC is equivalent to the *Mann-Whitney Test* or the *Wilcoxon Rank-Sum Test* (Krzanowski and Hand, 2009). It is a nonparametric alternative to the *Two-Sample t-test*, which uses the ranks of the data rather than their raw values to determining whether there

is a difference between two populations (in this case positive and negative instances). In a succinct way, AUC measures the power of discrimination for a binary classifier, by measuring how accurately it ranks the positive patterns before the negative patterns.

$$\text{AUC} = P(\text{random positive example} > \text{random negative example}) \tag{2.17}$$

When the AUC for a classifier is equal to one, it means that all positive and negative samples are correctly classified as so. Nevertheless, if the AUC is equal to zero it means that all the positive samples are classified as negative and all the negative as positive, i.e. our classifier perfectly discriminates the two classes but the decision is always wrong. In this case, if we exchange the label, the AUC will be equal to one. A pure random guessing means that the model does no discrimination at all between the two classes, and the AUC is equal to 0.5. However, no model should have an AUC less than 0.5. If that happens, it means that the model is able to discriminate between the two classes but it exchanges the labels, and to improve the model, the decision rule must be reverted.

Following the same theory of the ROC curves, AUC does not put more emphasis on one class over another, so it is not biased against the minority class. In this sense, we can say that it represents the performance averaged over all possible class distributions or cost ratios represented by the ROC curve.

Regarding the classifiers shown in Example 2.3, the final AUC value for each of them is obtained by averaging all the AUC values of all the curves obtained by cross-validation. The result is the following average AUC values for each classifier: svm1 with 0.943 ± 0.008; svm with 0.904 ± 0.009; nnet with 0.862 ± 0.015 and nnet1 with 0.835 ± 0.023).

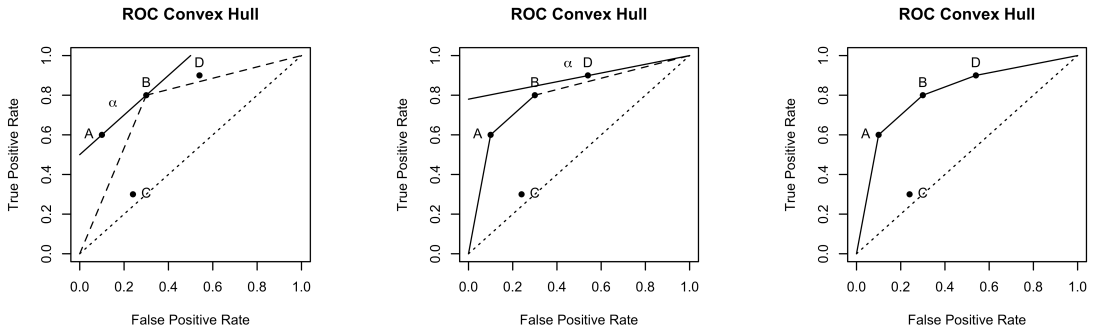
ROC Convex Hull (ROCCH)

The analysis of the ROC Convex Hull (ROCCH) allows to determine if a subset of classifiers is potentially optimal (cf. Provost and Fawcett (1997)). This analysis is specially important in scenarios with skewed distributions or cost-sensitive learning. Though these conditions may change, the ROC curve does not; what might change is the region of interest of the curve. Therefore, at different regions of the ROC space, which may correspond to different distribution skews, different classifiers can lead the performance. Through the ROCCH, the objective is to find out which are the classifiers potentially optimal across all the ROC space.

Having this goal, the convex hull [*2] of the points in the ROC space is drawn. Once we get the ROCCH, and according to Provost and Fawcett (2001); Provost et al. (1998), some conclusions can be drawn about the classifiers with operating points lying on the convex hull.

Any point in the ROC space that corresponds to a set of conditions, such as distribution/costs, can be mapped into a *iso-performance* line. That line is tangent to the point and establishes that all classifiers (points) with the same slope of the iso-performance line have, at most, the same performance. The flatter the slope of the iso-performance line, the higher is the TP-intercept. With the properties of these lines, it is possible to state that if a given point is on the convex hull then there is an iso-performance line tangent to that point such that there is no other line with the same slope that passes through another point and has a higher TP-intercept. Thus, no point lies above the final curve formed by the convex hull. In this context, the classifier represented by that point is optimal under the assumed distribution/costs indicated by that slope. Using this technique we are able to eliminate classifiers that are never optimal regardless of class distribution or misclassification costs.

Figure 2.6 exemplifies some of the choices made on the construction of the convex hull. In Figure 2.6 (a), we see that if the convex hull does not include the model A, then it is possible to have an iso-performance line tangent to A, which is above the convex hull and has a higher TP-intercept, i.e. a better performance. This "forces" the convex hull passing through A. A similar situation happens with model D in Figure 2.6 (b). In Figure 2.6 (c) we get to the final ROCCH. The only model we can eliminate, knowing that it will never be optimal, is model C.



(a) (b) (c)

Figure 2.6: The ROC Convex Hull (ROCCH) is a hybrid classifier made up from one or more classifiers.

Provost and Fawcett (2001) suggested that classifiers on the convex hull can be combined into a hybrid classifier. This results in a powerful classifier composed by a ordered sequence of classifiers.

[*2] The convex hull is the boundary of the minimal convex set containing a given non-empty finite set of points in the ROC space.

Pool Adjacent Violators (PAV)

Probability classifiers (e.g. Naive Bayes) assign to each example a score, i.e. a probability estimate of that example belonging to the positive class. The probability estimates can be used to rank instances from the most to the least likely to be a positive class instance. As we have referred, this kind of classifiers have their performance results heavily dependent on the *accuracy* of the probability estimates obtained. In this context, Zadrozny and Elkan (2001) proposed several methods to improve the calibration of probability estimates. According to Sheng and Ling (2006) no accurate estimation of probabilities is necessary if we obtain an accurate ranking of these estimates.

Zadrozny and Elkan (2002) proposed a calibration technique used to convert classification scores into more reliable probability estimates. This technique uses the Pool Adjacent Violators (PAV) (Fawcett and Niculescu-Mizil, 2007) algorithm to find new classifier scores through the minimization of the Brier Score [⋆3] of isotonic (monotonically increasing) regression.

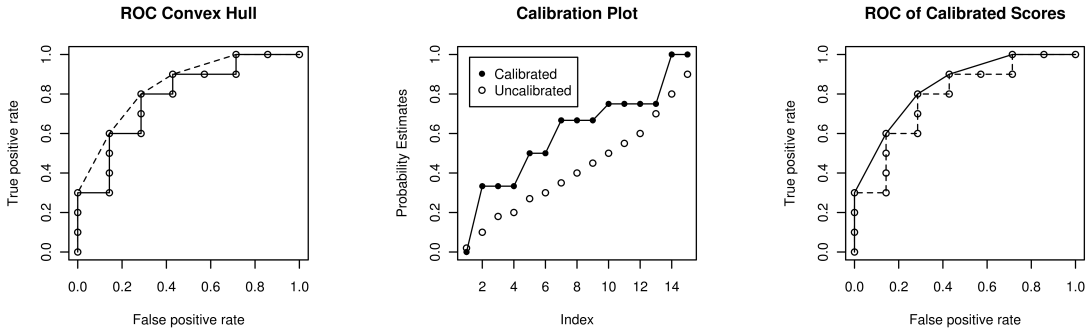
The PAV algorithm performs a monotonic increasing transformation of the probability estimates (i.e. scores) produced by the classifier. It starts by ordering the instances by the decreasing order of the scores given by the classifier and then assigns probability 1 to each positive instance and probability 0 to each negative instance. Then, it looks for adjacent violators, i.e. those contiguous instances whose scores violate the decreasing order after the probability has been assigned. When it finds one of such adjacent violations replaces their probabilities estimates by the average probability estimates of the instances involved. The process continues until it is obtained the initial sequence of instances with probabilities estimates monotonically decreasing.

A recent study (Fawcett and Niculescu-Mizil, 2007) has proved that PAV and ROCCH are equivalent. This means that generating the ROCCH of a single classifier is equivalent to finding the PAV-transformation of the scores of the original classifier. This process is illustrated in Figure 2.7.

In Figure 2.7 (a) we have an example ROCCH. On the following plot, we show the resulting calibrated scores from the application of the PAV algorithm to a set of original scores (cf. Figure 2.7 (b)). Finally, in Figure 2.7 (c), we have generated the ROC curve for the new calibrated scores. As it can be observed, the ROC curve is equivalent to the first generated ROCCH.

Calibrated scores are very important in decision-making in a cost-sensitive scenario, as they give more reliable estimates. Moreover, they also provide relevant insight about the algorithms performance in ROC analysis.

[⋆3] Brier Score is used to assess the quality of the probabilities predicted by a classifier (Brier, 1950). It is equivalent to the squared-error for binary class problems.



(a) (b) (c)

Figure 2.7: The ROCCH and the calibration of scores through the PAV algorithm result in the same ROC curve.

ROC Curves with Instance Varying Costs (ROCIV)

One of the criticisms made over ROC curves is their inability to deal with problems in which the error costs vary with the instances.

In a standard binary classification scenario, the expected cost of a classifier on a test set can be estimated as $C = FP \cdot c_{FP} + FN \cdot c_{FN}$, where c_{FP} and c_{FN} are fixed, and assumes that the class distribution is static, i.e. FP and FN are also fixed. As we have referred before, one of the advantages of the ROC curves is that they provide performance estimates independent of the class distribution or costs operating conditions. The iso-performance lines in the ROC space, presented by Provost and Fawcett (2001), represent the performance of classifiers for a specified operating condition: class skew and the misclassification error costs. Two given points in the ROC space, (FP_1, TP_1) and (FP_2, TP_2), have the same performance if they are in the same iso-performance line, which slope is defined by $m = \frac{FP_2 - FP_1}{TP_2 - TP_1} = \frac{c_{FP} \cdot p(N)}{c_{FN} \cdot p(P)}$, where p(N) and p(P) are the proportion of negative and positive examples, respectively. All the classifiers lying in the same iso-performance line are expected to have the same expected cost: $C = FPR \cdot p(N) \cdot c_{FP} + (1 - TPR) \cdot p(P) \cdot c_{FN}$.

Nevertheless, in some applications the cost is not uniform regarding each type of error, but varies with the instance. In those cases, the costs are usually determined by the value of the instance, and thus known exactly. In particular, and as we have already referred, cost-sensitive learning is one of techniques used to tackle the problem of prediction of rare classes. In Example 2.1 (page 12), we have shown a situation where the cost of an error may be dependent on the hypothetical fraud value involved.

In this context, Fawcett (2006b) has proposed ROCIV curves to cope with instance varying costs. To incorporate costs in the ROC graph it is necessary to transform the cost matrix into a cost-benefit matrix, where the costs are relative to making a positive prediction. In the ROCIV space,

the X-axis is the fraction of the maximum FP cost, and the Y-axis is the fraction the maximum TP benefits. AUCIV is obtained in a similar way to AUC and shares the same statistical properties, with the addition that instances are chosen in proportion to their costs. This feature makes ROC curves a technique applicable to a wider set of real-world applications, where the cost associated to different errors is not uniform.

2.3.2.3 Precision and Recall Analysis

Whenever the focus of the classification task is a specific target class, precision and recall are the two suitable performance measures. They estimate the performance concentrating on the target class and disregarding the performance over the other class.

Precision and recall were originally defined by Kent et al. (1955) in the area of *Information Retrieval*. Together, precision and recall evaluate the "quality" of retrieved results within the context of the information being searched for.

In the context of the prediction of rare classes, these measures can reveal how effective are the predictions of the model.

Definition 2.16 (**Precision**)**.** precision is the proportion of positive predicted instances that are effectively positive instances.

$$precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{2.18}$$

Definition 2.17 (**Recall**)**.** recall is the proportion of existent positive instances that are correctly predicted as so.

$$recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2.19}$$

Ideally, we would like a model with a high precision (FP $\rightarrow$ 0), but also with a high recall (FN $\rightarrow$ 0). Nevertheless, there is usually a trade-off between these two measures. recall is a non-decreasing function of the number positively predicted instances. This means that if all the examples are predicted as positive, a high recall is achieved. Still, one must be aware that if all the instances are predicted as positive and no discrimination from the negative examples is done, a poor precision is obtained.

Also in the field of Information Retrieval, Rijsbergen (1979), proposed an effectiveness measure E, presented in Equation 2.20, which is based on precision and recall.

$$E_\alpha = 1 - \frac{1}{\alpha \cdot \frac{1}{precision} + (1-\alpha) \cdot \frac{1}{recall}} \tag{2.20}$$

The α parameter, allows the specification of the relative difference of the importance of recall and precision.

The F-measure is given in Equation 2.21, by taking $\alpha = 1/(\beta^2 + 1)$,

Definition 2.18 (**F-measure**)**.** F-measure is a measure that evaluates the trade-off between precision and recall, by a weighted harmonic mean of their values.

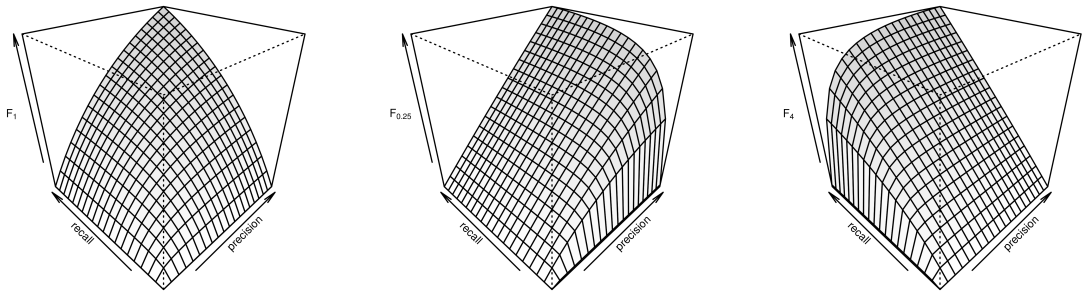
$$F = 1 - E_\beta = \frac{\left(\beta^2 + 1\right) \cdot precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{2.21}$$

where β, controls the relative importance of recall to precision

The F-measure assumes high values when both values of precision and recall are high. This means that low values of F are an indicator of poor performance. The β parameter is important, as its value has direct implications on the interpretation of F results.

- If $\beta = 1$ then the F-measure is the harmonic mean of precision and recall. In this case, it is common to denominate it as F.
- If $\beta \to 0$ then recall has less weight; in particular, when $\beta = 0$, the F-measure is the precision.
- If $\beta \to \infty$ then precision has less weight; in particular, for large values of β, the F-measure approaches the recall.

The graphs presented in Figure 2.8 show the F-measure obtained for three different values of β.



(a) F-measure with $\beta = 1$ (b) F-measure with $\beta = 0.25$ (c) F-measure with $\beta = 4$

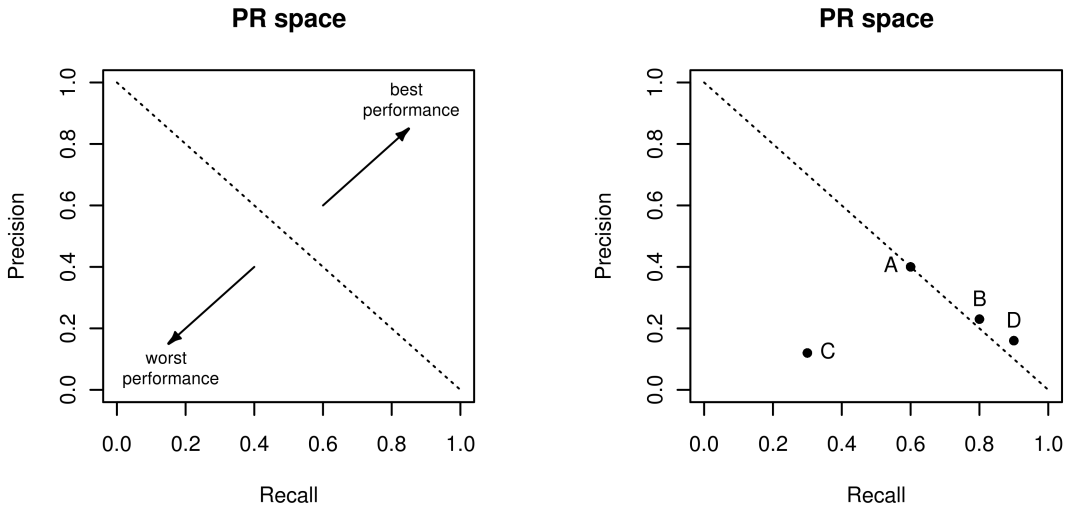
Figure 2.8: Different parameterizations for the F-measure.

The first graph 2.8 (a) shows the most usual situation, where $\beta = 1$. The second graph 2.8 (b) plots the situation where the precision is given 1/4 of the importance of recall, through the specification of $\beta = 0.25$. Finally, the third graph 2.8 (c) shows the situation where the recall is given four times the importance of precision, i.e. $\beta = 4$.

Precision-Recall Curves

Precision-Recall (PR) curves are specially useful when the notion of negative class is not so clear as the target positive class. These curves establish, in the so-called PR space, the relationship between recall (cf. Equation 2.19), and precision (cf. Equation 2.18).

Likewise in the ROC space, we can typically characterize a model by its position in the PR space, as shown in Figure 2.9 (a).



(a) PR Space Characterization

(b) Models Operating Points in the PR space

Figure 2.9: PR space characterization.

The ideal situation is to obtain the maximum value of both recall and precision and, thus, be near to the top right corner. The worst performing models are on the opposite corner, the bottom left corner. The remaining two corners depend on the trade-off we establish as acceptable between precision and recall.

In Figure 2.9 (b) we plotted the four models of Example 2.2 (page 21). It is interesting to observe that the models B and D, in comparison to their positions in the ROC space, do not have so good positions in the PR space. This has to do with the number of TN of model B and the number of FP of model D, as we will explain next.

ROC curves have shown to be a good visualization technique for the analysis of the performance of algorithms whenever there is an imbalance class distribution problem or error costs. Still, when dealing with highly skewed data sets, such as the case of the rare classes, they can give misleading estimates as argued by Drummond and Holte (2000, 2004).

Regarding the performance analysis of a classifier on a single target class, the results shown in the ROC space may be misleading. If a decrease in FPR occurs, that can be motivated by a decrease on FP, but also by an increase on TN. Therefore, if this last scenario is the true cause for the low FPR, an overly optimistic estimate is given for the model performance. Whenever there is an imbalanced class problem, one must be aware that the model will tend to make a low number of predictions of minority class POS, whether they are TP or FP.

In these cases of highly skewed data sets (Davis and Goadrich, 2006; Goadrich et al., 2006), PR curves give a more rigorous insight about the algorithm's performance. PR curves can expose differences between algorithms that may not be apparent in ROC space. precision is dependent on the prior distributions, as its definition depends on both classes. Thus it is directly related to the degree of skewing.

If the goal is to make the model evaluation sensitive to variations in binary class distributions problem, PR curves are the most appropriated tools. precision can detect the effect of large number of negative instances as it compares FP to TP.

Contrary to what happens with ROC curves, in PR space the two dimensions are not strongly correlated. A change in recall would not necessarily imply a change in precision, as there is no direct connection between FN and FP. As the recall increases or remains constant, it must be checked how it reflects in the value of the precision: it can decrease or remain constant too.

A typical behavior of a PR curve is shown in Figure 2.10. The final PR curve resulted from the interpolation of precision according to Equation 2.22. This rule reflects what we have just explained.

$$precision_{int}(r) = \underset{r' \geq r}{\operatorname{argmax}}\ precision(r') \tag{2.22}$$

Davis and Goadrich (2006) demonstrated that there is a relationship between PR curves and ROC curves by proving the following theorem.

Theorem 2.19. *For a given data set of positive and negative examples, there exists a one-to-one correspondence between a curve in ROC space and a curve in PR space, such that the curves contain exactly the same confusion matrices, if recall* $\neq 0$.

In effect, given that the number of positive examples is fixed for a certain curve (ROC or PR), it is possible to uniquely determine the confusion matrix associated to each point of the curve and derive the correspondent point in the other space (ROC or PR). In their work, Davis and

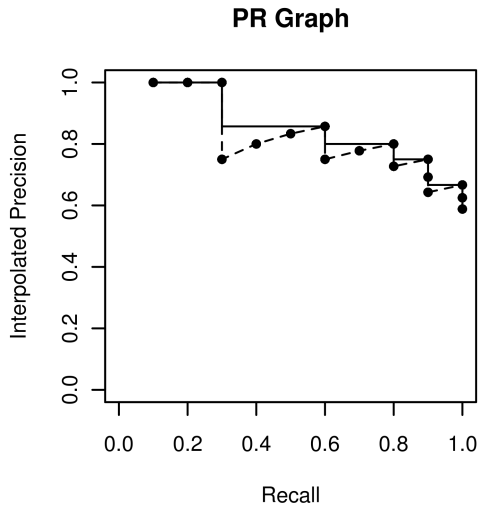


Figure 2.10: PR curve with interpolated precision.

Goadrich (2006), also proved that if a curve is dominant in one space, then it is also dominant in the other. Still, and as it would be expected, the ranking performance of other curves (which are not dominant) may change. Using ROC curves for imbalanced data may hide some difference among classifiers that becomes noticeable in the PR space.

One important measure to consider in the PR analysis is the generality, defined as follows.

Definition 2.20 (**Generality**)**.** The Generality (G) is the proportion of positive instances within all the existent instances and is defined by,

$$\mathrm{G} = \frac{\mathrm{POS}}{\mathrm{POS} + \mathrm{NEG}} = \frac{\mathrm{TP} + \mathrm{FN}}{\mathrm{TP} + \mathrm{FN} + \mathrm{TN} + \mathrm{FP}} \tag{2.23}$$

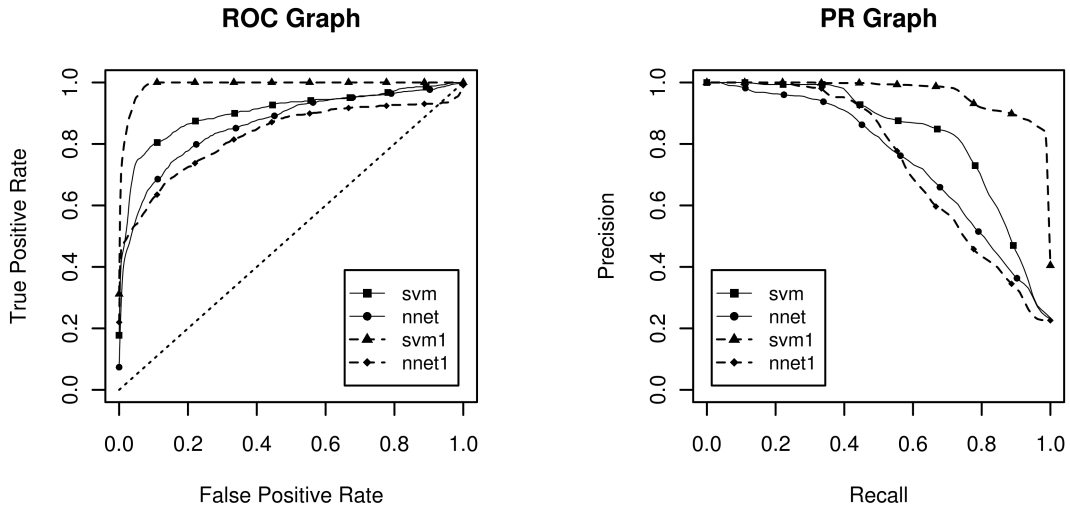
Example 2.4. PR curves: performance of classifiers of co-receptors used by HIV-1

In Figure 2.11 (b), we present the PR graph for the same models generated for the ROCR.hiv data set that was presented in the Example 2.3 (page 25).

In Figure 2.11 (a), we have the ROC curves shown previously. The corresponding PR curves are in Figure 2.11 (b).

As it is possible to observe, not all the differences among the classifiers estimatives of performance reflect in the same way in the two spaces. Even though the dominant model prevails as it is

expected (Davis and Goadrich, 2006), i.e. the svm1 is the best performance model in both ROC and PR spaces, the difference among the remaining three models does not remain the same. Namely, nnet1 is not the worst performing model anymore. In fact, for recall values below ≈ 0.55 *it shows better performance than the original nnet.*



(a) ROC graph

(b) PR graph

Figure 2.11: ROC and PR curves of four different models. The relative positioning of the models' curves is not the same in both graphs.

Area Under the PR Curve (AUC-PR)

Examining the entire PR curve is very informative, like in ROC curves, but if we want to proceed to a more formal statistical analysis, it is necessary to summarize this information through, for example, the *Area Under Curve for PR curves* (AUC-PR). However, as there is no linear relationship between precision and recall, calculating AUC-PR involves two previous steps (Goadrich et al., 2006):

i. standardize the PR curve so to cover the full range of recall values $[0, 1]$;

ii. perform the interpolation between the points which comprise the PR curve.

Regarding the first step, the first point of the curve (r_1, p_1) is extended horizontally to $(0, p_1)$. This last point is surely achievable, as we can discard any fraction of retrieved positive instances and still

get the same precision on the remaining instances. This is also a result of Equation 2.22. Regarding the point for the maximum recall value, in the worst case, if the model predicts everything to be positive, the point $(1, \mathrm{G})$ can always be found.

The interpolation in the PR space is not linear and thus not straightforward as in ROC space. Though every point in the PR curve is generated from the underlying TP and FP counts, to interpolate two points in the PR space we need to know the local skew between them. Assuming that we have points A and B with their respective $\mathrm{TP_A}$, $\mathrm{TP_B}$, $\mathrm{FP_A}$ and $\mathrm{FP_B}$ counts, their local skew is defined by $\frac{\mathrm{FP_B}-\mathrm{FP_A}}{\mathrm{TP_B}-\mathrm{TP_A}}$, which gives us how many negative examples it takes to equal one positive example. By increasing the value of x, such that $1 \leq \mathrm{x} \leq \mathrm{TP_B} - \mathrm{TP_A}$, i.e. $\mathrm{TP_A}+1, \mathrm{TP_A}+2, \ldots, \mathrm{TP_B}-1$ the resulting interpolation points are given by,

$$\left(\frac{\mathrm{TP_A} + \mathrm{x}}{\mathrm{POS}}, \frac{\mathrm{TP_A} + \mathrm{x}}{\mathrm{TP_A} + \mathrm{x} + \frac{\mathrm{FP_B}-\mathrm{FP_A}}{\mathrm{TP_B}-\mathrm{TP_A}}\mathrm{x}} \right) \tag{2.24}$$

From this interpolation process, new points are created and the AUC-PR can be approximated.

Like in ROC curves, the optimal AUC-PR is one, but for a random classifier is not 0.5. The reason is the fact that the interpolation is not linear. Thus for those random classifiers AUC-PR is equal to Generality (G).

Achievable PR Curve (ACHPR)

Analogous to the convex hull in ROC curves, in PR curves there is the *achievable PR curve*. The method (Davis and Goadrich, 2006) to obtain such a curve consists in:

i. find the convex hull in ROC space;

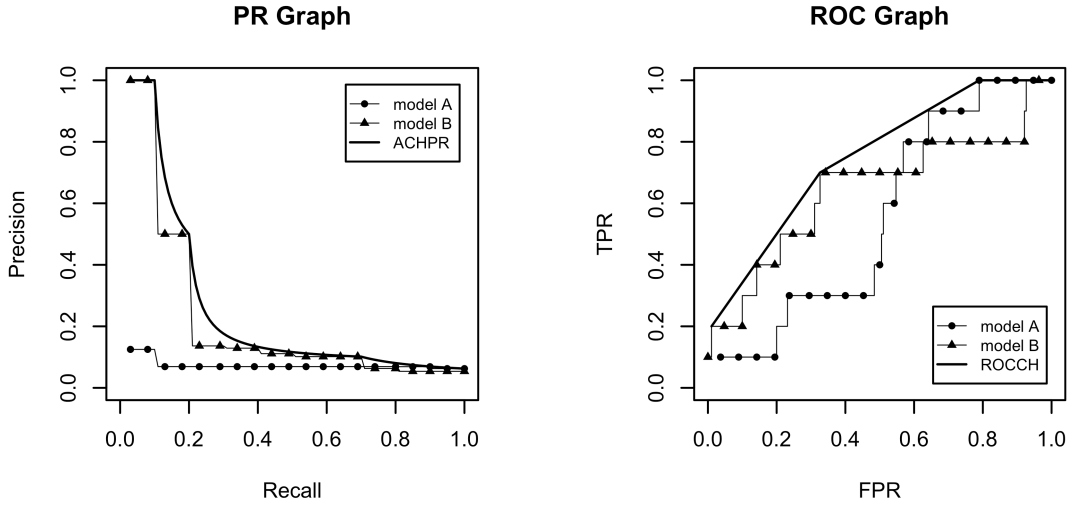
ii. for each point to be included in the hull, use the confusion matrix to derive the corresponding point in PR space;

iii. perform the interpolation between the newly created PR points.

Example 2.5. Achievable PR curve: two synthetic models.

*We conceived two synthetic models, A and B, with an imbalanced class distribution (*G = 0.1*). The objective is to illustrate the obtained ACHPR through the steps mentioned by Davis and Goadrich (2006).*

In Figure 2.12 (a), we plot the PR curves of the two models. In order to get the achievable PR curve*, we move to the ROC space and find the ROCCH shown in Figure 2.12 (b). Finally, we move back to the PR space, where we convert the ROCCH into the final ACHPR, performing the necessary interpolations. The result is shown in Figure 2.12 (a).*



(a) The ACHPR for Model A and B

(b) The ROCCH for Model A and Model B

Figure 2.12: An Achievable PR Curve (ACHPR).

The ACHPR has an AUC-PR of ≈ 0.282*, whereas the ROCCH has an AUC-ROC of* ≈ 0.748.

Based on conditions similar to the ones shown in the Figure 2.12, Davis and Goadrich (2006) claim that algorithms that optimize the AUC-ROC can not be guaranteed to optimize the AUC-PR. In this concrete example, while there is no dominant model in the ROC space, the same does not happen in the PR space, where model B dominates for almost all the values of recall.

2.3.2.4 Discussion on Performance Measures

The F-measure is the most commonly used single-value performance measure involving precision and recall. Nonetheless, besides the F-measure, other performance measures have been conceived combining precision and recall. We will present some of the measures (cf. Manning et al. (2008)) that rely on PR curves, next.

Definition 2.21 (**PRBEP**)**.** Given a PR curve, the Precision/Recall Break-Even Point (PRBEP) is the value at which the precision is equal to the recall.

Definition 2.22 (**Prec@k**)**.** The Prec@k is the precision value calculated after k predictions were made.

This measure has the advantage of not requiring any estimate of the number of positive instances existing in the data set. But, at the same time, it is unstable as the number of positive instances occurring in the k predictions has a strong influence.

Definition 2.23 (**R-precision**)**.** The R-precision is the precision value measured at a fixed recall value R.

R-precision overcomes the previous problem of Prec@k by adjusting the calculation of precision to the existing positive instances in the data set.

Definition 2.24 (**11-AP**)**.** *11-point Average Precision* (11-AP) is the result of averaging the precision values of the curve at the 11 fixed recall levels: $0.0, 0.1, 0.2, \ldots, 1.0$, as shown in Figure 2.13.

Definition 2.25 (**MAP**)**.** The *Mean Average Precision* (MAP) is the mean of the average precision in several curves, obtained for a set of pre-specified levels of recall (e.g. 11-AP). It is obtained as follows (cf. Equation 2.25),

$$\text{MAP} = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{M_j}\sum_{k=1}^{M_j} precision\,(r_{jk}) \tag{2.25}$$

where N is the number of recall levels and M_j is the number of iterations (curves) from which to do the average at each recall level.

Nonetheless, all these measures are more indicated for those situations where the end-user establishes how many TP are likely to be found. As argued by Goadrich et al. (2006), AUC-PR has advantages over the PRBEP, as the PR curves are not necessarily generated from ranked

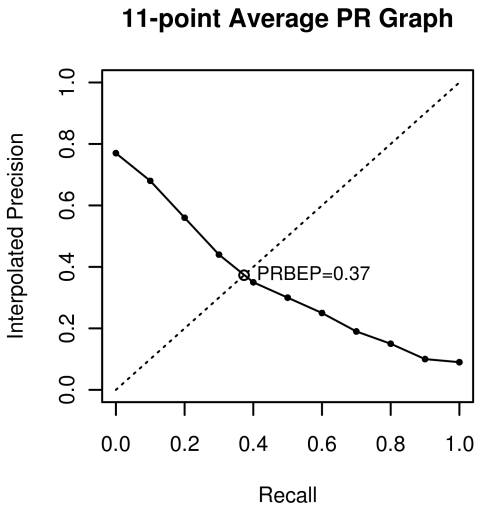


Figure 2.13: Average 11-point PR graph.

list of retrieved examples. It can also be more exact than 11-AP, as for AUC-PR all the points for all the range of recall are calculated and interpolated.

The overview of evaluation measures and visualization tools mentioned in this section is necessarily incomplete. For instance, regarding visualization tools we could also have mentioned the Lift Charts (Witten and Frank, 2005) and Cost Curves (Drummond and Holte, 2006). Lift Charts are able to represent the advantage of using a prediction model over a small portion of the data set. Cost Curves share many of the properties of ROC curves, but bring additional features such as an easier and reliable selection of the best algorithm by visualizing the statistical significance of the difference in performance of two classifiers.

The decision on which visualization tool to use is domain-dependent. For instance, in medical applications ROC curves are widely accepted and understood by the end users. In applications where class probabilities are unknown or highly variable, being largely skewed, *precision-recall* analysis is more appropriate.

If the interest is quantifying the "quality" of a classifier, then scalar performance measures should be used, such as F. All the measures relate to a single decision threshold, or operating point. This would be perfect in an environment where class priors probabilities and misclassification costs are known. On the contrary, if we want to obtain ranking information about the performance of different classifiers, than ranking-aware measures should be used, such as AUC-ROC, AUC-PR, MAP and PRBEP. These are also single-number characterizations. They can be used to assert some statistical significance of the performance ranking.

More recently, regarding the AUC measures, it came up a controversy around the fact of their evaluation being trustful. Hand (2009) argues that, even though AUC is a widely used measure for evaluating the performance of classifiers it does shows some drawbacks. Besides the known problem of giving potential misleading estimates if the ROC curves cross, it also varies the evaluation according to the classifier. This means that the AUC derived from the curve depends on the empirical distribution of the scores given by the classifier itself. In this sense, Hand argues that if there are differences among the classifiers they depend on their empirical score distributions. Thus, the measure AUC reflects the evaluation of different classifiers using different measures. The misclassification costs should not depend on the scores it produces. In this context, and unless the cost ratio distribution is specified, Hand (2009) proposes the H measure.

Definition 2.26 (H**-measure**)**.** The H-measure is a standardised measure of the expected minimum loss obtained for a cost balance c established between the two classes, chosen from a random distribution.

According to Hand, this measure allows for fair comparisons to be made, independent of the scores distributions, and based on what is called a *belief distribution* that reflects the application goals, namely based on the class priors. Hand (2009) suggests choosing c from the *beta distribution* with the shape parameters $\alpha = 2$ and $\beta = 2$, a binomial symmetric cost distribution (cf. Figure 2.14).

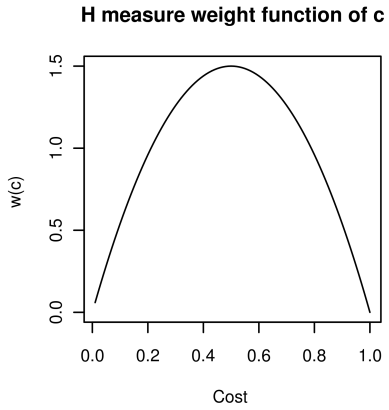


Figure 2.14: Beta Distribution used by the H-measure.

2.3.2.5 Biased Supervised Learning Algorithms

In this section we discuss some of the existing approaches for the imbalance class problem, and namely rare class prediction, that make use of some of the techniques we have seen so far.

Several studies (e.g. Weiss and Provost, 2001) have been conducted about the influence that train data class distribution has on learning. Some results based on ROC analysis have shown that

the "natural" distribution is not always the best distribution for training, specially for highly imbalanced class distributions. Therefore, sampling techniques have become the most usual way of addressing class rarity in machine learning. Their goal is to make the learning task easier by re-balancing the class distribution. Nevertheless, the basic and more intuitive techniques have some drawbacks. In this sense, new and more advanced sampling techniques have appeared in order to make them more effective regarding the class imbalance problem. In this subsection, some of these techniques will be described.

The basic sampling techniques consist on a simple heuristic sampling towards the balance of class distribution: under-sampling and over-sampling.

The under-sampling method, randomly eliminates examples of the majority class. The major problem with this approach is that we may incur on the risk of eliminating important majority class examples for the discrimination process of the classes.

Similarly, the over-sampling method, randomly re-samples examples of the minority class. Again, it presents a major problem of overfitting by just replicating information, transmitting the apparent impression of *accuracy* in the train data. Both of these techniques suffer from lack of other sources of information and their results are easily surpassed by more sophisticated sampling methods.

Kubat and Matwin (1997) improved under-sampling by removing the majority class examples that are considered redundant. These examples are the ones belonging to the majority class that are far away from the decision border. The authors also eliminate the examples identified as noise or lying in the borderline of the decision. Nevertheless, the identification of such set of examples is very time consuming for large data sets. Moreover, one can never ensure that only the proper examples are removed.

Having in mind the influence that class distribution has on learning, Chan and Stolfo (1998) focused on obtaining the best class distribution for a given problem by generating multiple data sets where an ensemble of classifiers would then be learned and aggregated into one classifier. Each data set is generated by joining the minority class with a subset of the majority class ensuring that all majority class examples are used at the end. This gives origin to a set of well-balanced data sets. The problem with this approach is the time complexity involved in the preliminary experiments necessary to achieve the best distribution.

Boosting (Freund and Schapire, 1997) was developed with the intention of addressing a given difficult classification task by iteratively learning weak classifiers with respect to a distribution and adding them to a final strong classifier. One of the most popular applications of this technique is on imbalance class problems through the AdaBoost algorithm, which was presented by Fan et al..

AdaBoost is a boosting algorithm, which assigns different weights on the training distribution of each iteration. Examples that are correctly predicted have its weight decreased for the next iteration. On the contrary, examples wrongly predicted have its weight increased in the following iteration. The objective is to obtain a more powerful model at the end, which is able to learn even the "harder" examples, as it is the case of examples of the rare class.

AdaCost was then proposed by Fan et al. (1999) as a small extension of AdaBoost. The particularity about AdaCost, relative to AdaBoost, is the cost-sensitivity of the weights' updating rule. Examples of the minority class that are misclassified are assigned higher weights in the next iteration, than the rest of the misclassified examples.

RareBoost (Joshi et al., 2001) is another boosting algorithm whose weight-update rule is determined by the perception of the differences between FP, TP and TN on each iteration. The argument of the authors is that AdaCost may sometimes over-emphasize recall, thus leading to poorer precision.

The SMOTE algorithm, proposed by Chawla et al. (2002), generates synthetic minority examples to over-sample the minority class, without replicating examples. Each new example results from an interpolation between some randomly chosen examples of k-nearest (usually k=5) neighbors minority examples. This way it opens up the instance space of the minority class, without causing overfitting. In fact, results shown in Chawla et al. (2002) confirm that a better generalization of the minority class is achieved.

Chawla et al. (2003) argued in their work that both boosting and over-sampling can overfit, as it weights examples belonging to the minority class more than those belonging to the majority class. In this context, they propose SMOTEBoost, which alters the distribution not by updating the weights associated to each example, but by introducing new minority-class examples using the SMOTE algorithm. Results have shown that SMOTEBoost have achieved higher F values then AdaCost.

Costing, proposed by Zadrozny et al. (2003), uses "cost-proportionate rejection sampling" to avoid overfitting. In particular, each instance in the original training set is sampled once, and accepted into the sample with a probability proportional to the misclassification cost. For a specific parametrization, the sample can contain all the examples of the minority class and sampling without replacement of the majority class. After this sampling, the algorithm performs bagging. It allows the use of an arbitrary cost-insensitive learning algorithm as a black box in order to accomplish cost-sensitive learning.

Estabrooks et al. (2004) claim, based on their experiments, that a classifier formed as the combination of classifiers induced with under-sampling or over-sampling at different rates, achieves

better results in comparison to just one classifier. Even though it is not clear what is the best sampling technique to use and with which parametrization, the result of their study lets us conclude that, in general, the combination of techniques seems to outperform the use of one method only.

On a comparative study conducted with several methods over several data sets with imbalanced class distribution, Batista et al. (2004) proposed variations of SMOTE that use some of the under-sampling techniques used in Kubat and Matwin (1997). Results have shown that these new methods, which comprise over-sampling with a data-cleaning under-sampling, presented good performance results in highly skewed data sets.

According to Weiss (2004), all the available data should be used, regardless of the class distribution. Cost-sensitive learning algorithms should be preferable to sampling techniques. However, if there is a very high lack of data, or the data is very expensive to obtain, then sampling techniques must be used. To truly assess the performance over the rare class, proper evaluation measures should be used.

Still, research studies on the poor performance of the learning systems on imbalance class problems continue. Some researchers (Jo and Japkowicz, 2004; Prati et al., 2004) argue that it is not the imbalance of the classes itself the responsible for the bad performance of the learning system, but the presence of the *small-disjuncts* in the models.

In conclusion, in what sampling techniques are concerned to tackle imbalanced class distribution problems, or even rare classes problems, there is no unified solution. Even though some relevant results have emerged, the research field remains active.

Different bias can be introduced into the learning algorithms so that pre-specified goals can be successfully accomplished. Some algorithms optimize the AUC of the ROC curve, others precision and recall. The objective is to improve the performance of the algorithms under imprecise environments, such as the ones with imbalance class distributions or different error costs. In fact, there is a wide range of classifiers biased to the problem of imbalanced class distribution and misclassification costs. We are going to mention just a few as example.

PNRule is a two-phase rule induction algorithm, proposed by Joshi et al. (2002), which copes with the concepts of precision and recall. During the first phase, it learns the so-called P-rules which predict the presence of the minority (positive) class. During this phase, achieving a good recall is the objective. In the second phase, it learns the N-rules that predict the non-existence of the minority class. The objective in this phase is to improve the precision, by eliminating the FP included in the first phase. According to the authors, PNRule has shown good results on problems with rare classes where there is a strong correlation between the rules that distinguish the minority class from the majority class.

Ferri and Flach (2002) proposed a new decision tree algorithm, designed for imprecise environments. This decision tree learner has as new features, the ability of representing a set of classifiers by relabelling its leaves and using the AUC as split criterion. Experiments have shown that its *accuracy* performance with these new features were satisfactory.

Prati and Flach presented ROCCER, a rule selection algorithm guided by the ROCCH, thus optimizing the AUC. Experiments have shown that ROCCER tends to select smaller rule sets than other rule selection algorithms, but with comparable performance results.

Nevertheless, it should be noticed that, according to Davis and Goadrich (2006), algorithms that optimize AUC-ROC do not optimize AUC-PR. For this reason, and even though the algorithms that optimize AUC-ROC show a different bias in comparison to traditional classification algorithms, they do not share the same objectives of the algorithms for the prediction of rare classes.

2.3.3 Unsupervised Approaches

Even though most of the research on Utility-based Data Mining is focused in classification tasks and it is highly related to cost-sensitive learning, it is important to have in mind that it does not confine to it. Utility-based Data Mining is a wide area of learning, which aims to be closer to real world scenarios and even though most of the existing work in Utility-based Data Mining is associated to prediction tasks, utility has also impact on unsupervised learning tasks, such as clustering. Weiss et al. (2008) actually claim additional research for these tasks, arguing that the same considerations on utility theory hold. Unsupervised outlier detection is one of such tasks.

In the machine learning field, several techniques for outlier detection appeared in the area of unsupervised learning. In effect, due to the the lack of labeled and known instances of outliers this constitutes a wide area of research concerning outlier detection. These techniques do not have any premise on previous information, they only assume that for some similarity measure, outliers will appear isolated or in very small groups.

The choice of the method depends on the number of dimensions of the data, data type, sample size, algorithms efficiency (time-complexity and space- complexity), and the user understanding of the problem.

Due to space limitations we will just briefly describe a few unsupervised outlier detection algorithms. Following the taxonomy presented by some researchers (e.g. Chandola et al., 2009; Chen et al., 2010; Zhang et al., 2007), we outline statistical methods, clustering methods, distance-based and density-based methods and finally, those attached to more complex data sets, spatial, spatial-temporal and sequential detection methods.

2.3.3.1 Statistical Methods

Statistical techniques were the first ones used in outlier detection. The goal of these techniques is to check if the data follows an assumed theoretic distribution. All the points that satisfy a statistical discordance test for the assumed distribution are declared outliers. In that sense, an outlier is an object that has a low probability with respect to a probability distribution model of the data.

The more traditional approach is the identification of univariate outliers under the premise of a *Normal* distribution for a continuous random variable which is assumed to be identically and independently distributed. The *Normal* distribution is fully determined by two parameters: mean (μ) and variance (σ^2). The probability density function, also designed as *Gaussian curve*, is a bell-shaped function that gives the probability that the random variable takes a value in a given interval. If the sample values of the variable follow the *Normal* distribution then the obtained *Gaussian curve* is symmetric around μ, meaning that the probability of having a value lower than μ is equal to the one of having a value higher than μ, i.e. 50%. Graphically, we also see a violation to the symmetry assumption as an indicator of the presence of outliers: breakdowns on the curve, too heavy or too short tails. The probability density curves shown in the Figure 2.15 indicate the presence of outliers in all the three hypothetical variables.

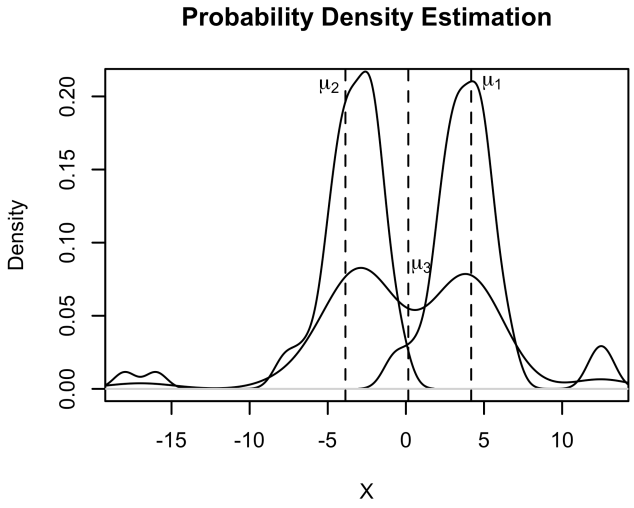


Figure 2.15: Probability density function of random continuous variables.

To find outliers, we need to examine how the values spread around the central value of the distribution. One of the methods used for this purpose is the *Grubbs' Test* (Grubbs, 1969), also called ESD method (*Extreme Studentized Deviate*). This method calculates the z-score (cf. Equation 2.26) for each observation.

$$z = \frac{|x - \mu|}{\sigma} \tag{2.26}$$

The value of z gives the distance from the point x to the mean μ in units of the standard deviation σ. The variable z follows a *Standard Normal* distribution (having mean 0 and standard deviation 1). If the obtained z value is large enough, i.e. greater than a critical value known from the z-table (for instance, greater than two for, approximately, 5% significance level), then it is considered an outlier.

Based on this method Rousseeuw and Leroy (1987) established a distribution-based definition of outlier, as presented in Definition 2.27.

Definition 2.27 (**Distribution-based Univariate Outlier**)**.** Let Y be a set of observations from a univariate Normal distribution $N(\mu, \sigma)$ and p a point from Y. Then the z-score for p is greater then pre-specified threshold if and only if p is an outlier.

Nevertheless, this method is not fully reliable for outlier detection as it is based on measures that are themselves heavily influenced by the presence of outliers, namely the mean and the standard deviation, which may fail the identification of some outliers.

A more informal but robust approach is given through the box plot introduced by Tukey (1977) which does not assume any distribution, but emphasizes the tails of the distribution. The box plot (e.g. Cleveland, 1993) provides a visual summary of important aspects of a distribution, namely symmetry and skewness, because it depends on two so-called robust statistics: the median and the inter-quartile range. The designation robust comes from the fact they provide estimates less influenced by outliers.

The box plot consists of a box with two edges, one on each side of the box. The box itself contains the middle 50% of the data as it is delimited by the first quartile (Q_1), which indicates the 25^{th} percentile, and the third quartile (Q_3), which indicates the 75^{th} percentile. The range of the middle two quartiles is known as the inter-quartile range (IQR). The edges that extend outside the box, are called whiskers. Essentially, each whisker extends as far as the most extreme observation, which is not classified as a potential outlier. The adjacent values are the thresholds that determine the existence of outliers.

The box plot points out two thresholds for potential outliers, which are obtained based on the data's Q_1 and Q_3. According to Tukey's definition, these thresholds - the adjacent values - are the two most extreme values in the data within 1.5 times the inter-quartile range ($IQR = Q_3 - Q_1$) away from the first and third quartiles, as described below.

Definition 2.28 (**Box Plot Outlier**)**.** Let Y be a set of observations, Q_1 the corresponding first quartile, Q_3 the third quartile, IQR the inter-quartile range, adj_L the low adjacent value and adj_H the high adjacent value.

The set of outliers O is defined as follows,

$$O := \{y \in Y \mid y < adj_L \ \vee \ y > adj_H\} \tag{2.27}$$

where

$$adj_L = \max\{y \in Y \mid y \leq Q_1 - 1.5 \cdot IQR\} \tag{2.28}$$

$$adj_H = \min\{y \in Y \mid y \geq Q_3 + 1.5 \cdot IQR\} \tag{2.29}$$

The set O can be further split and defined into one of two categories:

- Mild Outliers (MO)

$$MO := \{y \in O \mid y \leq Q_1 - 1.5 \cdot IQR \vee y \geq Q_3 + 1.5 \cdot IQR\} \tag{2.30}$$

- Extreme Outliers (EO)

$$EO := \{y \in O \mid y \leq Q_1 - 3 \cdot IQR \vee y \geq Q_3 + 3 \cdot IQR\} \tag{2.31}$$

The Figure 2.16 displays a box plot where the main thresholds are signaled.

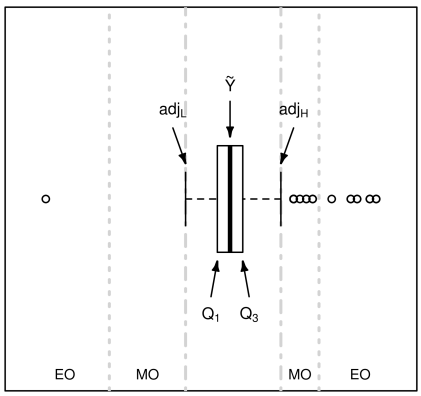


Figure 2.16: The box plot provides five statistics that characterize the data: the median ($\tilde{Y}$), the 1^{st} and the 3^{rd} quartiles (Q_1 and Q_3) and the two adjacent values (adj_L and adj_H). With this information, the set of outliers is defined by its extremeness as mild outliers (MO) or extreme outliers (EO).

The next example illustrates how the presence of outliers can influence the detection of univariate outliers through z-scores and the box plot rule.

Example 2.6. Univariate Outlier: detection of Ca outlier values in soil samples.

The Condroz data (Rousseeuw et al., 2008) is a set of 428 observations which contains the pH-value and the Calcium (Ca) content in soil samples, collected in different communities of the Condroz region in Belgium. All the data used in this study has a pH-value between 7.0 and 7.5.

Figure 2.17 (a) is the plot of Ca measurements. From the plot it is possible to visually identify the presence of some outlier Ca values within the soil samples. We then try to identify the outlier soil samples according to the Grubbs' Test *with the z-score, as shown in Figure 2.17 (b), and Tukey's definition of a box plot, as shown in Figure 2.17 (c). The outliers detected by the first method were signaled on plot (a) as black points. From the comparison of the graphs (b) and (c) we find that a very different number of observations is identified as outlier. The reason is that Ca has only high extreme values, which bias the mean to higher values and diminishes the z-value for some of the potential outliers. The box plot, on the contrary, does not suffer such influence as the thresholds for the outliers are established upon quartiles of the data.*

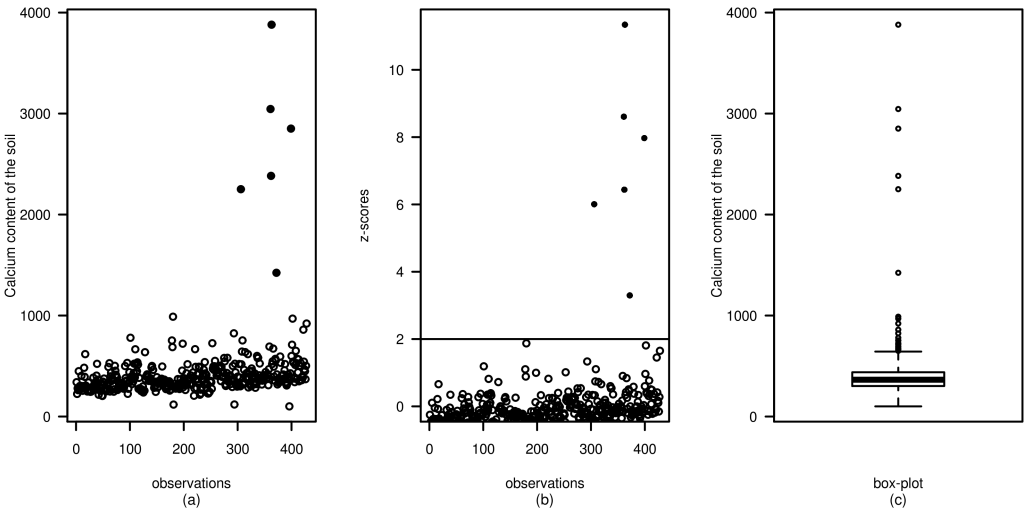


Figure 2.17: Identification of univariate outliers by the z-score and by the box plot.

Extreme Values

In the context of what we have said so far, extreme values are potential outlier candidates. For many years extreme values motivated curiosity in several sciences such as ecological, meteorological, economics, physics and engineering. In these areas, it is typical to observe skewed distributions, i.e. heavy tails on the probability density distribution which indicates the presence of not only rare but also high or low extreme values. Tukey had already drawn the attention for this fact.

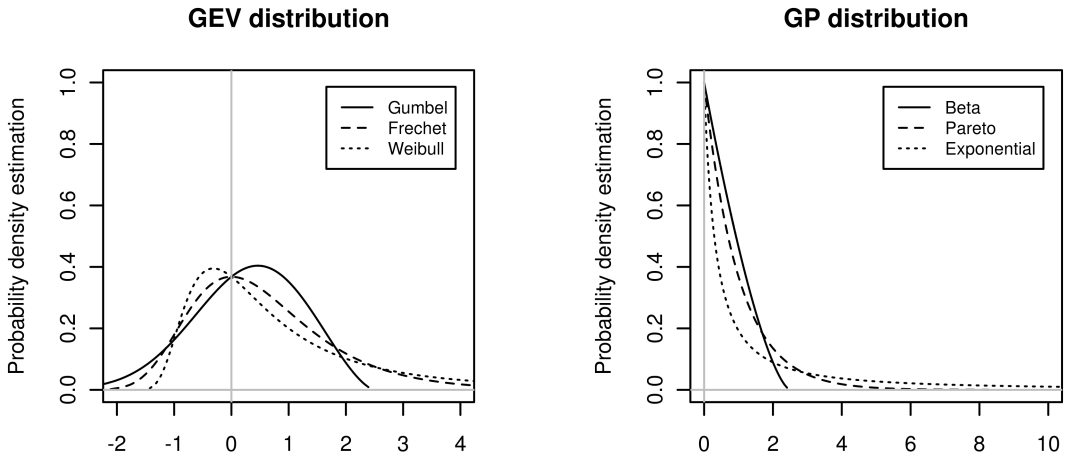
> "As I am sure almost every geophysicist knows, distributions of actual errors and fluctuations have much more straggling extreme values than would correspond to the magic bell-shaped distribution of Gauss and Laplace."
>
> John Tukey

The importance of effective risk management on areas such as financial markets gave origin to the *Extreme Value Theory* (EVT) (Coles, 2001; Embrechts et al., 1997, 1998), which is now an important field of research in statistics. It is focused in finding probabilistic models able to extrapolate and estimate values outside the range of existing data.

One of the proposals of EVT is the method Block Maxima. According to this model, the extreme (maximum or minimum) of a sequence of observations, under very general conditions, is approximately distributed as the *Generalized Extreme Value* (GEV) distribution. This distribution can only be described by one of the three models: *Gumbel*, *Fréchet* and *Weibull* distributions. As it is exemplified by Figure 2.18 (a), for an example of only high extreme values, each of these three distributions have different characteristics. The *Gumbel* distribution has a light upper tail and is positively skewed. *Fréchet* distribution has heavy upper tail and infinite higher order moments. Finally, the *Weibull* distribution has a bounded upper tail. Informally speaking, *Fréchet* distribution is a heavy-tailed distribution; *Gumbel* distribution is light-tailed; and, finally, *Weibull* distribution is bounded, i.e. with finite tails.

The *peaks over threshold* (POT) method was proposed later and it is used to estimate the excess distribution with respect to a threshold level u. This model follows a *Generalized Pareto* (GP) distribution as a good approximation to the tails of the distribution, i.e. observations exceeding the specified high threshold, described by EVT. The GP distribution has one of the three forms: *Beta*, *Pareto* and *Exponential*, all illustrated in Figure 2.18 (b).

The three models of GP distribution are similar to the ones defined for GEV distribution. In this context, the *Pareto* distribution is a heavy-tailed distribution (sometimes called "power law");



(a) *Generalized Extreme Value* (GEV) Distribution

(b) *Generalized Pareto* (GP) Distribution

Figure 2.18: Different distributions in *Extreme Values Theory*.

the *Exponential* distribution is a light-tailed distribution; and, finally, the *Beta* distribution is a bounded distribution.

The application of EVT has shown useful results in hydrology, climatology and finance. Still, depending on the application, if operational risk of capital is involved, convergence of the estimates provided by methods based on EVT to a theoretic maximum (or minimum) can be slower than real data. In those cases, alternative simulation techniques have to be used (Degen et al., 2007).

Multivariate outlier detection is not as straightforward as it is univariate outlier detection, because multivariate distributions have no tails. Multivariate outliers are cases that have an unusual combination of values for a number of variables. The value for any of the individual variables may not be a univariate outlier, but in combination with other variables compose an outlier. Multivariate outliers are, sometimes, also univariate outliers.

The univariate outlier detection technique we have seen is based on the distance of a value to the central value of the distribution considered - the mean if the *Normal* distribution is chosen.

The basis for multivariate outlier detection is the *Mahalanobis* distance which also assumes a *Normal* distribution. This measure calculates the distance of every multivariate example $x = (x_1, x_2, \ldots, x_p)$ to the mean of the data set $\mu = (\mu_1, \mu_2, \ldots, \mu_p)$ using the covariance matrix Σ, as it is shown in Equation 2.32.

$$D_M(x) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} \tag{2.32}$$

The square of this distance is distributed as χ^2_d distribution, i.e. a chi-square distribution with d-degrees of freedom. Within this distribution, a threshold value is proposed for the outliers detection, according to a significance level (usually 1% or 5%).

Definition 2.29 (**Distribution-based Multivariate Outlier**)**.** Let $\mathcal{DS}$ be a set of d-dimensional multivariate observations from a normal distribution and p a point in $\mathcal{DS}$. The square of the *Mahalanobis* distance follows a χ^2_d distribution with d-degrees of freedom. Then the *Mahalanobis* distance is larger then a pre-specified threshold if and only if p is a multivariate outlier.

This distance incorporates dependencies between attributes by the covariance matrix and, at the same time, is independent of the variables' scale as it standardizes them to a mean of zero and a variance of one.

Unfortunately, similarly to z-score, *Mahalanobis* distance is itself affected by outliers. The estimates of the mean and covariance matrix are extremely sensitive to the presence of outliers.

In this context, and so that more candidate outliers can be detected, robust multivariate estimators have appeared. Rousseeuw (1985) proposed a robust estimate for the covariance matrix, which is based on weighted observations according to their distance from the center. Some other robust estimators include *Minimum Covariance Determinant* (MCD) and the *Minimum Volume Ellipsoid* (MVE) (Rousseeuw, 1985; Rousseeuw and Leroy, 1987).

To illustrate the influence of outliers in a multivariate scenario, we present the following example within a medical application.

Example 2.7. Multivariate outliers: detection of a heart disease severe risk.

In this example we show how the identification of multivariate outliers helps the diagnostic of individuals who suffer potential severe risks of a heart disease, such as extremely high diabetes and cholesterol or other symptoms. The data set diabetes(Faraway, 2008) is the result of a study conducted over 403 African Americans to understand the prevalence of obesity, diabetes, and other cardiovascular risk factors in central Virginia. For each individual 19 variables were measured, such as age, gender, height, weight, waist, Total Cholesterol, Stabilized Blood Glucose, High Density Lipoprotein (HDL), and Glycosylated Hemoglobin, among others. For our outlier identification, we removed all the cases with missing values, which left us with 130 observations.

Some domain knowledge was used to visually inspect those individuals who were at clear life risk. In this context, we decided to plot two of the most informative measured variables:

- *Glycosylated hemoglobin:* a value greater than 7% is taken as a positive diagnosis of diabetes;
- *Total Cholesterol:* a value greater than or equal to 240 mg/dL is considered a high risk of developing heart disease.

If you have diabetes, you're more likely to have more cholesterol abnormalities — which highly contributes to a cardiovascular disease.

From the observation of Figure 2.19 (a) we can realize that the measured Glycosylated Hemoglobin evidences the presence of outliers, by extreme high values.

In Figure 2.19 (b) and (c) we show two methods for the identification of outliers. The first one, shows to the Mahalanobis *distance and identifies the five individuals signaled with a black point in the plot (a). The second one, uses a robust version of the* Mahalanobis *distance that identifies more outliers. This result is quite interesting as we realize that, with the second method, we are also able to identify those individuals that, even though do not have the maximum value of Stabilized Blood Glucose and Total Cholesterol, should be in a precaution area. Hypertension is another important risk factor for the development and worsening of many complications of diabetes. Therefore, instead of this set of attributes we could have chosen to analyze diabetes regarding the values of blood pressure, sex and age, for instance.*

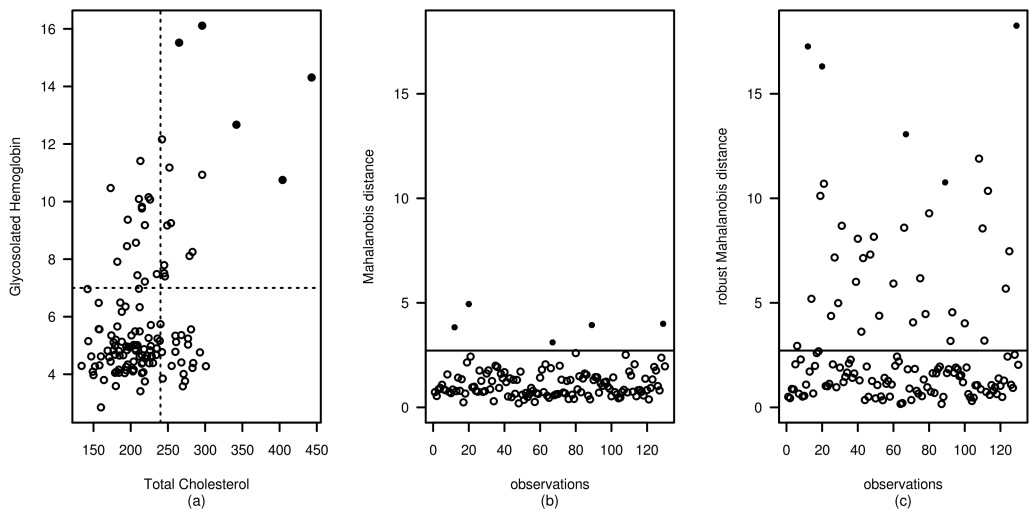


Figure 2.19: Identification of multivariate outliers through the *Mahalanobis* distance and the robust *Mahalanobis* distance.

Laurikkala et al. (2000) have shown that the threshold definition of outliers used by the box plot is also extendable to multivariate outliers with real-valued, ordinal and categorical attributes. The

scatter plot, such as the one shown in Figure 2.19 (a), is a graphical technique which can be used to detect outliers in two-dimensional data sets. In these plots an outlier is a point that deviates significantly from a linear model between the two variables. Other plots exist for more than two-dimensional data sets (Zhang et al., 2007), but for high dimensional data the outlier detection can be very time consuming and an inaccurate process.

Most of the statistical methods, present some limitations as argued in Hodge and Austin (2004). All are parametric, i.e. attached to a known theoretic distribution, which requires prior knowledge about the data for accurate results to be obtained. In many situations, data sets do not really follow any particular distribution. In fact, data can be of any type and is usually described by a large number of variables.

In complex problems it is often hard to decide what is the theoretic distribution of the normal instances. For these situations non-parametric methods are used. These methods do not make any assumption on the functional form of the probability distribution. This function is estimated from the data using, for instance *kernel* functions. But these methods are computationally expensive, especially for high-dimensional data sets. Estimating the parameters of multidimensional distributions is harder and often inaccurate.

Classic statistical methods go beyond the scope of our study. Further details can be found in the literature (e.g. Coles, 2001; Embrechts et al., 1997; Petrie and Sabin, 2005).

2.3.3.2 Clustering Methods

Cluster analysis is a common technique with the objective of group data instances into groups based on a similarity measure. Even though in a indirect way, this technique is used for outlier detection. The reasoning is that the clustering methods group normal instances together in one or more dense clusters, while outliers are isolated in low density clusters or do not belong to any cluster at all. Following this theory, there are several clustering based outlier detection techniques. In this section, we will mention some of them.

BIRCH (Zhang et al., 1996) was one of the first algorithms coping with outliers notion. For Zhang et al. (1996) all the points lying in less dense regions were interpreted as potential outliers. The clustering algorithm identifies as potential outliers those points which were not included in none of the formed clusters.

Definition 2.30 (**Clustering-based Outlier**)**.** Outliers are points that do not belong to clusters of a data set or belong to clusters that are significantly smaller than other clusters.

Another cluster-based outlier detection was recently proposed by Torgo (2007), which takes advantage of the resistance each point offers in the aggregation step of an hierarchical clustering, as a way to measure its *outlyingness*.

2.3.3.3 Distance-based Methods

Nearest neighbour analysis is a widely known and used technique in data mining in which a data object is analysed in the light of its nearest neighbours, i.e. with respect to the data objects that are closer to it. This type of analysis is data-driven, which means that, contrary to the statistical approaches, no assumption is made upon the underlying distribution of data.

Regarding outlier detection, distance-based methods were originally proposed by Knorr and Ng (1998). The basic idea is that we should look for outliers in a locally sparse area within the instance space.

There are three popular definitions of outlier, from a distance-based detection perspective. The first one was proposed by Knorr and Ng (1998) as follows,

Definition 2.31 (**Distance-based Outlier - I**)**.** A point p in data set $\mathcal{DS}$ is a DB(f,D)-outlier if at least a fraction f of points in $\mathcal{DS}$ lies at a distance greater than D from p.

This definition posed some difficulties, namely in the determination of the distance D. Ramaswamy et al. (2000) extended further this definition so that outliers can be more efficiently discovered and ranked. Their proposal was based on the distance to the k^{th} nearest neighbour of a point D^k.

Definition 2.32 (**Distance-based Outlier -II**)**.** Given k and n, a point p is an outlier if no more than $n-1$ other points in the data set have higher D^k value than p. The top n points with the maximum D^k values are the D^k_n outliers of the data set.

Outliers are the top n examples whose distance to the k^{th} nearest neighbour is the greatest. Still, according to Angiulli and Pizzuti (2002), this definition fails for only considering the distance to its k^{th} neighbour, ignoring information about the closer points. Such a scenario is exemplified in Figure 2.20) for two points q_1 and q_2 and a neighbourhood of eight points ($k = 8$).

According to the Definition 2.32, q_1 and q_2, we would get $D^8(q_1) = D^8(q_2)$, i.e. would be ranked equally, which is absurd. In this context, another definition was recently suggested by Angiulli and Pizzuti (2005),

Definition 2.33 (**Distance-based Outlier - III**)**.** Given a data set $\mathcal{DS}$ and integer a k, the weight w of a point p is defined as the sum or the average of the distances separating it from its k neighbours. Out^n_k is the set of top n outliers of $\mathcal{DS}$ with respect to k, i.e. the points which score the largest n values of weight.

One of the problems with distance-based approaches is scalability of the algorithm with large data sets. Some efforts have been made (e.g. Ghoting et al., 2008) with the goal of finding fast algorithms for this task, but they rely on several assumptions that limit their application.

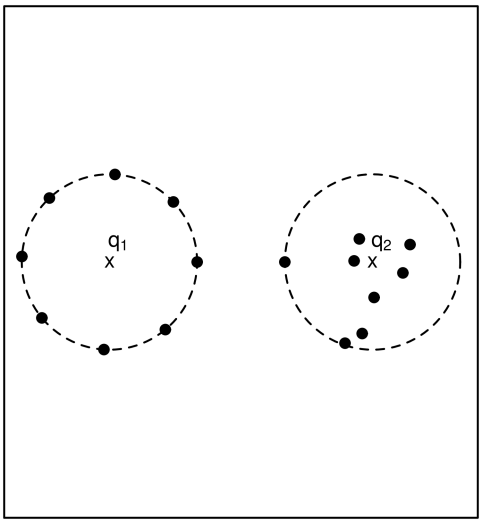


Figure 2.20: Two different points q_1 and q_2 that would be wrongly labeled as "equal" outliers (given that $D^8(q_1) = D^8(q_2)$) by the distance-based detection method.

2.3.3.4 Density-based Methods

Breunig et al. (2000) proposed an algorithm specially designed for outlier detection, claiming that the concept of outlier should be locally inspected. This approach was motivated by some limitations found in distance-based outlier detection when dealing with data sets with both sparse and dense areas, such as the one shown in Figure 2.21. From the visual inspection of the figure, we identify two clusters C_1 and C_2 of different densities. Regarding outliers, o_1 is easily detected, but for o_2 to be detected as outlier by the distance-based definition, all the points of C_2 would have to be outliers too.

The weak performance of distance-based approaches in data sets with a non-uniform density, lead to the formulation of a density-based approach and to the definition of the Local Outlier Factor (LOF) (Breunig et al., 2000) method. The essence of the approach is that being an outlier is not a binary property, is a characteristic that every object has, but with some score assigned.

The outlier factor is used to rank the objects regarding their *outlyingness*. The outliers are no longer treated equally. Every point that lies outside a cluster should be identified as outlier, but each one is given a score, LOF, which captures the relative degree of isolation of the point from its surrounding neighbourhood. According to Breunig et al. (2000), points that are deep inside a cluster have approximately a LOF of 1. The other points are assigned a value within a tight interval, regardless of whether the considered nearest neighbourhood embraces one or more clusters. Therefore, the local outlier factor of a point is the reciprocal of the density in the point's neighbourhood. This lead to a new outlier definition,

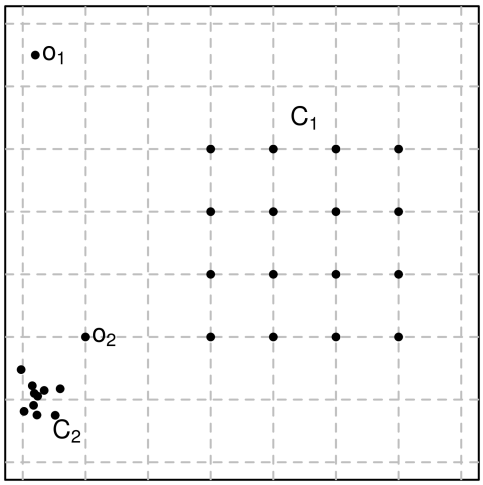


Figure 2.21: One of the main limitations of distance-based detection of outliers is being density careless.

Definition 2.34 (**Density-based Local Outlier**)**.** Outliers are points that lie in the lower local density areas with respect to the density of its local neighbourhood.

To better illustrate how this new density-based local outlier definition addresses some of the limitations shown by the distance-based outlier definitions, we return to the instance space shown in Figure 2.21. In the Figure 2.22 are represented the subset of instances which compose C_2 and the point o_2, which could not be identified as outlier in the previous distance-based approach.

In the same figure are also pictured two distance concepts introduced by (Breunig et al., 1999):

- *core-distance* of a point p - $core(p)$ - is its distance to its k^{th} nearest-neighbour;
- *reachability-distance* between two points p_1 and p_2 - $r(p_1, p_2)$ - is the maximum between the *core-distance* and their actual distance.

In a elementary way, the reasoning is that high values of *reachability-distance* between two given points indicates that they may not be in the same cluster. The *local reachability-distance* of a point is defined to be inversely proportional to the average *reachability-distance* of its k neighbourhood. By the formula presented by the authors, LOF assigns high values to the points that have a much lower *local reachability-distance* in comparison to its k neighbourhood.

In these conditions, and considering the instance space of Figure 2.21, to the point o_2 would be assigned a higher LOF then to the points belonging to the clusters C_2 or C_1.

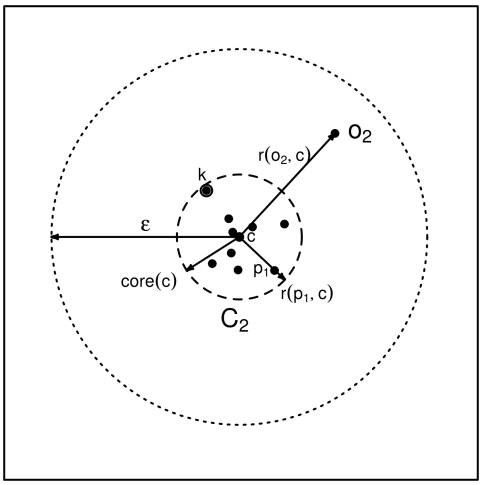


Figure 2.22: Illustration of local reachability distance of points relative to a core-distance of a point c.

Again, some limitations have been detected in this methodology. Namely, the selection of the neighbourhood (k), which is not always trivial, and the time complexity of the algorithm, which results from the fact of density being itself defined in terms of distance.

LOCI was one of the proposals (Papadimitriou et al., 2003) to fasten the LOF algorithm. One of its features is a different definition of local neighbourhood, based on a radius, not on a fixed number of points.

Later, another efficient density-based outlier detection method for large data sets was proposed (Ren et al., 2004): RDF-based outlier detection, where RDF stands for relative density factor. The relative density of a point p is given by the ratio of the density of the point and the average density of its neighbours. The outlier score of a point becomes its relative density. This leads to another density-based definition of outlier,

Definition 2.35 (**Relative Density-based Outlier**)**.** A point can be considered as an outlier if its own density is relatively lower than its nearby high density pattern cluster, or its own density is relatively higher than its nearby low density pattern regularity.

This method uses a vertical data structure (*P-trees*) to efficiently index data and prune the points which are deep in clusters, and then detects outliers only within the remaining small subset of the data, using the RDF.

In summary, density-based approaches have the ability of detecting local outliers that other methods may not be able to detect.

2.3.3.5 Spatial Methods

These methods are closely related to clustering methods. They are often used in geographical information systems (GIS) and in practical applications such as spread of diseases and climate studies (e.g. Shekhar et al., 2003).

The main distinguishing feature of spatial data is the existence of spatial attributes and the neighbourhood relationship.

Shekhar et al. (2003) describes the definition of outlier in a spatial context as follows,

Definition 2.36 (**Spatial based Outlier**)**.** A spatial outlier is a spatial referenced point whose non-spatial attribute values are significantly different from those of other spatially referenced points in its spatial neighbourhood.

Spatial outliers can be further split into two categories (cf Shekhar et al. (2003)): multidimensional space-based outliers and graph-based outliers. Their difference relies on their neighbourhoods. For the multidimensional space-based outliers, the neighbourhood is defined with the *Euclidean* distance and for the graph-based outliers, the neighbourhood is defined through graph connectivity. Methods for detecting outliers in a multidimensional *Euclidean* space have several limitations, because, as we have mentioned before, they assume an isometric space. Shekhar et al. (2003) propose the notion of neighbourhood outlier in a graph structured data set which has shown to be an efficient outlier detection technique for spatial outliers, namely in comparison to clustering techniques.

In several geographical applications we have to consider temporal correlations on top of spatial relationships. Therefore, there exists a spatial-temporal relationship that should not be neglected, when detecting outliers in these applications. Cheng and Li (2006) defined spatial-temporal outliers, as follows,

Definition 2.37 (**Spatial-Temporal based Outlier**)**.** A spatial-temporal outlier is a point whose non-spatial attribute values are significantly different from those of other spatially and temporally referenced points in its spatial or/and temporal neighbourhoods.

2.3.3.6 Sequential and Data Streams Methods

In many applications, data is a set of symbolic sequences. This is the case of the composition of the DNA that is a sequence of symbols belonging to the set {A,G,C,T}, surveillance systems,

network intrusion and so on. In these settings, the outliers may point out some surprising and/or suspicious activities in a monitoring process.

For distance-based or density-based techniques, it is very difficult to define the notion of similarity between the sequences. One of the alternatives is to use tree-based approaches. Sun et al. (2006) present the Probabilistic Suffix Tree (PST) data structure that employs the theory of *Markov Chain*. This structure enables the efficient calculation of similarity between a sequence and a set $\mathcal{S}$. Outliers, in the context of this type of data, can be defined as follows,

Definition 2.38 (**Sequence-based Outlier**)**.** Given a sequence p and a set $\mathcal{S}$, it is said that p is an outlier with respect to $\mathcal{S}$ if their value of similarity is lower than a user-defined threshold.

In some contexts, large volumes of data are continuously arriving in an ordered sequence, in what is usually known as a data stream. Data streams are infinite processes consisting of data that continuously evolves with time. As the memory is not infinite, the data must be stored in a more compact way and if necessary be expunged. In this scenario, a model built in a particular instant may not be valid in the near future.

In many data stream applications, only the most recent elements are important for data mining, while the old ones have a small or null importance. For example, in a stream of stock market data, traders are more concerned about the recent movements of the stock price. This may lead to focusing the analysis on the most recent N values of the stream.

Muthukrishnan et al. (2004) proposed the following definition of outlier based on the data summarization by histograms.

Definition 2.39 (**Data-Stream Outlier**)**.** If the removal of a point from the time sequence results in a sequence that can be represented more briefly than the original one, then the point is a data-stream outlier.

Outlier detection over data streams is mainly based on statistical techniques or density estimates. In spite of being a recent area, several algorithms have been proposed. Algorithms for detecting top-k monitoring outliers (Li and Han, 2007), for on-line detection of outliers over sensor streams data (Sheng et al., 2007; Subramaniam et al., 2006), are examples of this type of work.

This area has been receiving a lot of attention as the notion of outlier in data streams is associated with change-point detection in time and thus, related to fraud detection, rare event discovery, trend change detection and more.

2.4 Discussion

Most existing work on Utility-based Data Mining (UBDM) is related to supervised learning, namely classification tasks. Utility is associated to costs/benefits in a cost-sensitive learning setting.

Utility as a concept metric is application dependent. In some domains utility is associated to the rarity of a phenomenon. The detection or prediction of rare cases, also called outliers, are of high interest in situations such as ecological catastrophes, frauds, financial markets and so on. However, as we have seen, handling rare cases in data mining implies different methodologies than the ones traditionally used, which are biased to the detection / prediction of common cases.

Regarding supervised outlier detection, namely classification, we have seen that more appropriate evaluation measures are required so that a more reliable evaluation of the models is performed. Moreover, different learning techniques are also necessary in order to build classification models that are effective in the prediction of such rare classes.

With respect to unsupervised outlier detection, and given the wide range of real-life applications, extensive research has been performed. There are currently several approaches suitable according to application requirements.

While a lot of research has been carried out in utility-based classification and in outlier detection, regression is being almost neglected in this setting. One of the main contributions of this thesis is to extend utility-based learning to regression, namely in the prediction of rare and extreme values.

- 3 -

Utility-based Regression

"Nothing can have value without being an object of utility."
Karl Marx (1818 - 1883)

Utility-based learning is a key technique for addressing many real world data mining applications. Most of the existing research has been focused on classification problems. In this chapter, we present a methodology for handling regression problems in the context of applications with non-uniform costs and benefits across the domain of the continuous target variable. We propose utility-based metrics that calculate the cost/benefit of a prediction made by some regression model. These evaluation metrics provide a more informed assessment of the utility of any regression model given the application-specific preference biases. Within this setting, it is more reliable to compare and select alternative regression models according to the user/application requirements. Throughout this chapter, we will use a real-life application to better illustrate the objective of our proposed utility-based evaluation methodology. The referred application is the prediction of outdoor air pollution by measurements of NO_2 concentration values. This is one of the typical prediction tasks where non-uniform costs are required. The experiments on this data set show the advantages of our utility-based evaluation in the context of this class of regression problems. In order to better evaluate our claims, we also describe a set of diverse artificial regression problems with non-uniform benefits and costs to prove that the effectiveness of the results obtained by our utility-based metrics are not application dependent.

3.1 Introduction

The genesis of utility theory (Neumann and Morgenstern, 1944) emerged in an economic context as a way to express people's preferences in a numerically useful way. An utility function $u\colon X \to \mathbb{R}$ is used to represent the preference relation $\preceq$ on X, such that for every $x, y \in X$, $u(x) \leq u(y)$ implies $x \preceq y$. Originally, these preferences were associated to the game theory in an economic context. In such context, the goal for an investor is to maximize the utility based on the risk associated with

a set of alternative decisions/actions for a certain scenario. Given the estimated wealth and the probability associated to each alternative action, the investor uses an utility function to map the wealth into a utility score. Based on this function, the investor selects the action that yields the highest utility score. Currently, there are several utility functions that map a wealth value into a utility score proposed in the literature (e.g. Friedman and Sandow, 2010).

In this chapter we propose a methodology for utility-based regression tasks. In regression problems we have a potentially infinite set of possible decisions (the predictions). Any regression model estimates, for each test case, the value with highest probability and uses this value as the prediction (i.e. decision). What we claim is that for certain applications, this decision should take into account the wealth of each possible prediction. With this goal we propose methods for estimating the wealth of any possible prediction of a regression model. This means that, within our target applications, we will assume that the utility of each prediction is actually equal to its wealth. Hence, we will use the identity utility function ($u(w) = w$) in all illustrations of our methodology in this thesis. In this context, and to simplify the used terminology we will call our proposal an estimate of the utility of a prediction. We propose to estimate the utility (wealth) of a prediction (i.e. a possible action) by calculating its costs and benefits.

In many real world applications the costs/benefits associated to predictions are not uniform. Typically, these applications are related to cost-sensitive decision problems, where different predictions can lead to different decisions involving costs/benefits (e.g. credit approval, insurance contracts, targeted marketing). These applications have motivated the work on cost-sensitive learning (e.g. Domingos, 1999; Elkan, 2001) and, more recently, on utility-based mining (e.g. Weiss et al., 2005, 2008; Zadrozny et al., 2006). However, most of these studies focus on classification tasks (e.g. Drummond and Holte, 2000; Fan et al., 1999; Provost et al., 1998), even though similar problems arise within regression tasks.

In effect, there are several real world applications where the continuous target variable shows non-uniform costs across its domain. The anticipation of a critical phenomenon in domains such as finance, meteorology, ecology and fraud detection, are among this kind of applications. In these domains, the critical phenomena are usually described by a particular subset of values of a continuous variable, which usually triggers some sort of alarm or action. Thus, predictions made for this subset of values should have a differentiated cost/benefit to be in accordance with the application goals.

As mentioned by Crone et al. (2005), the majority of the work in regression assumes uniform costs. The estimated performance of a regression model is given by an average statistic over the magnitude of all the prediction errors. To overcome the limitations of this assumption several authors (Christoffersen and Diebold, 1996; Crone et al., 2005) have proposed new loss functions

for regression. Nevertheless, the proposed functions are only capable of distinguishing under-predictions from over-predictions, i.e. situations where the predicted values are below or above the true values, respectively. As we will illustrate later on this chapter, these loss functions, as well as other alternative evaluation measures, do not fully address the requirements of many important real world applications with non-uniform costs/benefits.

How can we evaluate, compare or select regression models in the context of applications with non-uniform costs/benefits? Moreover, how can we make possible to express the preference bias of the users of these applications? In this chapter, our objective is to answer these questions. We propose a methodology for evaluating regression models in the context of arbitrarily shaped cost/benefit functions across the domain of the numeric target variable of regression tasks. This methodology enables the creation of new evaluation metrics that incorporate the notion of utility and thus are able to provide better feedback on the merits of the regression models in the context of specific biases of any numeric prediction task. We will illustrate the use of our proposed metrics and perform some experimental validation of those metrics on real and artificial data.

3.2 Problem Formulation

In a predictive learning problem the goal is to learn the model that better fits a target concept based on a training set. The training set is defined by $\mathcal{DS}$ where x_i is a feature vector from the feature space $\mathcal{X}$ composed by the predictor (independent) variables and y_i is an instance of the target (dependent) variable Y with domain $\mathcal{Y}$. Depending on the domain of the target variable, we say that we have a classification problem (if $\mathcal{Y}$ is discrete); or a regression problem (if $\mathcal{Y}$ is continuous). The objective is to learn the best approximation of an unknown function $f : \mathcal{X} \rightarrow \mathcal{Y}$. The obtained approximation, $\hat{f}_\Omega$ is a model with a set of parameters, Ω, which are obtained by optimizing a preference criterion. In classification, the preference criterion commonly used to obtain $\hat{f}_\Omega$ is the minimization of the expected error rate (cf. Equation 2.1 - page 9). In regression, the standard preference criteria employed are also based on error estimates, such as the absolute error or the squared error.

The expected absolute error loss of a regression model $\hat{f}_\Omega$ with respect to a set of instances $\langle \mathrm{x}, y \rangle \in \mathcal{X} \times \mathcal{Y}$ drawn from a distribution D is defined as

$$E_{AE}(\hat{f}_\Omega, D) \ := \ E_{\langle \mathrm{x}, y \rangle \sim D} \ [\ |\hat{f}_\Omega(\mathrm{x}) - y| \ ] \tag{3.1}$$

Similarly, the expected squared error loss of a regression model $\hat{f}_\Omega$ with respect to a set of instances $\langle \mathrm{x}, y \rangle \in \mathcal{X} \times \mathcal{Y}$ drawn from a distribution D is defined as

$$E_{SE}(\hat{f}_\Omega, D) \ := \ E_{\langle \mathrm{x}, y \rangle \sim D} \ [(\hat{f}_\Omega(\mathrm{x}) - y)^2] \tag{3.2}$$

For all the error estimates, the objective is to find the model that minimizes the expected loss. In this context, and assuming that the squared error loss is the criterion, the optimal prediction $\hat{y} \in \mathcal{Y}$ for a given example $\mathrm{x} \in \mathcal{X}$ is determined by,

$$\hat{y} \ = \ \underset{y \ \in \ \mathcal{Y}}{\operatorname{argmin}} \, E_{SE}(\hat{f}_\Omega, \langle \mathrm{x}, y \rangle) \tag{3.3}$$

In standard regression setups like the ones presented above, the usefulness of a prediction is inversely proportional to its loss value, and thus to the difference between true and predicted values. In this context, one can define the following principle.

Definition 3.1 (**Principle of Utility in Standard Regression**)**.** In standard regression scenarios, utility is a function of the error of the prediction (i.e. $L(\hat{y}, y)$).

Definition 3.2 (**Properties of Utility in Standard Regression**)**.** In general, and assuming that L is a regression loss function (e.g. absolute error) and U is an utility function, we have that the utility function is inversely proportional to the loss function, i.e. $U \propto L^{-1}$. Given any two pairs of predictions $\langle \hat{y}_1, y_1 \rangle$, $\langle \hat{y}_2, y_2 \rangle \in \mathcal{Y} \times \mathcal{Y}$ the following properties hold in these setups.

i) **Equal accuracy predictions have the same usefulness.**

$$L(\hat{y}_1, y_1) = L(\hat{y}_2, y_2) \implies U(\hat{y}_1, y_1) = U(\hat{y}_2, y_2)$$

ii) **More accurate predictions are always preferable.**

$$L(\hat{y}_1, y_1) < L(\hat{y}_2, y_2) \implies U(\hat{y}_1, y_1) > U(\hat{y}_2, y_2)$$

For the applications we target with our proposal, these otherwise reasonable properties, can be counter-intuitive. As a consequence, and similarly to the conclusions drawn in classification (e.g. Elkan, 2001; Fan et al., 1999), the standard error measures demonstrate to be non effective for applications with non-uniform costs/benefits.

3.2.1 An Application: Prediction of Outdoor Air Pollution

Nowadays, road traffic, industries and forest fires are among the main factors that affect the quality of the air that we breed, specially in big cities. The exposure to such air pollution leads to adverse health effects. Therefore, the increasing percentage of unhealthy substances has determined the establishment of concentration limits for some toxic composts.

The World Health Organization (WHO) has established the so-called AQGs (*"Air Quality Guidelines"*) with the purpose of controlling the air quality based on the potential risk to human health. The AQGs include the air quality standards, i.e. permissible levels for the six most commonly found air pollutants. The referred pollutants are particle pollution (PM), ground-level ozone (O_3), carbon monoxide (CO), sulfur dioxides (SO_2), nitrogen oxides (NO_x), and lead (Pb) [*1]. Keeping track of the concentration values of these pollutants is highly important, namely because the exposure to them is associated with numerous effects on human health, including increased respiratory symptoms, hospitalization for heart or lung diseases and even lung cancer which may lead to premature death.

Nowadays, there is a broader consciousness about keeping the concentration of the pollutants at levels that protect public health and the environment. As such, the air quality standards have been changing over the recent years making the annual average concentrations stricter, in order to reduce the emission of the pollutants.

The concentration of the pollutants varies both on systematic factors (e.g. car traffic volume) and on meteorological factors (e.g. wind speed and direction, precipitation, air humidity, temperature). In areas of high motor vehicle traffic, such as large cities, the amount of nitrogen oxides (NO_x) emitted into the atmosphere can be quite significant. In the presence of excessive oxygen (O_2), NO_x will be transformed into the more harmful nitrogen dioxide (NO_2), which is a major air pollutant that is toxic when inhaled.

Nitrogen dioxide is an important atmospheric gas, not only because of its bad health effects but also because it absorbs visible solar radiation and contributes to the constitution of smog. In this context, it can also have a potential role in global climate change if its concentration becomes too high. Therefore, the control of concentration values of NO_2 is a very important task.

As described in the WHO (2006) report and re-established in the last directive - Directive 2008/50/EC - of the European Parliament and of the Council of May, 21^{st} of 2008 on ambient air quality and cleaner air for Europe [*2] - limits shown in Table 3.1 are the current thresholds imposed for NO_2 concentration.

[*1] Detailed information on how these pollutants can affect the air quality can be found in WHO (2006).

[*2] All the EU Legislation, and in particular the Directive 2008/50/EC, can be accessed through EUR-Lex (`http://eur-lex.europa.eu/`)

Table 3.1: NO_2 hourly concentration thresholds for the protection of human health.

NO_2 **Hourly Thresholds defined according to the Directive 2008/50/EC**	
Limit Value:	$200\mu g/m^3$;† Margin of Tolerance: 50%; Deadline by which limit value is to be achieved: 01.Jan.2010;
Upper assessment threshold:	$< 70\%$ of limit value;†
Lower assessment threshold:	$> 50\%$ of limit value;†
Alert threshold:	$400\mu g/m^3$;

† Not to be exceeded more than 18 times a calendar year.

According to this directive, a combination of monitoring measurements and modelling techniques may be used to assess ambient air quality if the level of the pollutant in question is between the lower and upper assessment thresholds.

For illustration purposes on predicting the outdoor air pollution, we have used real data from a study (Aldrin and Haff, 2005) that relates the air pollution in a road with the traffic volume and meteorology. The data *[3] consists of a subsample of 500 observations from a data set, collected by the Norwegian Public Roads Administration.

The target variable (`LNO2`) are the log-transformed concentration values of NO_2 measured in $\mu g/m^3$ on each hour, at Alnabru in Oslo, Norway, between October 2001 and August 2003. The predictor variables are factors that influence the concentration: the logarithm of the number of cars per hour; the temperature registered 2 meters above ground; the wind speed, the temperature difference between 25 and 2 meters above the ground; the wind direction in degrees; the hour of day; and day number from October 1, 2001. A more detailed description of the data can also be found in Aldrin and Haff (2005).

According to the directive, and even though this is known to be a complex system where factors such as the meteorology interfere, one of the objectives is that by 2010 the hourly average concentration of NO_2 did not exceed $150\mu g/m^3$ for more than 18 times per year.

In Figure 3.1, we have the probability density function (pdf) of the `LNO2` variable. As we can observe, almost all registered values are below the limit value of $200\mu g/m^3$ ($\ln(200\mu g/m^3) \approx 5.3$). We should pay special attention to all the values that are above the limit value. These are the values that should be avoided, or that should occur no more than 18 times a year. In 2010, the objective was to use $\ln(150\mu g/m^3) \approx 5.0$ as the limit value.

In this context, it is perceptible that the *relevance* of the values of the target variable `LNO2` is *not uniform*. Higher concentration values, near $\ln(150\mu g/m^3)$ or above, are more important than

*[3] Available on the StatLib Datasets Archive: `http://lib.stat.cmu.edu/datasets/`

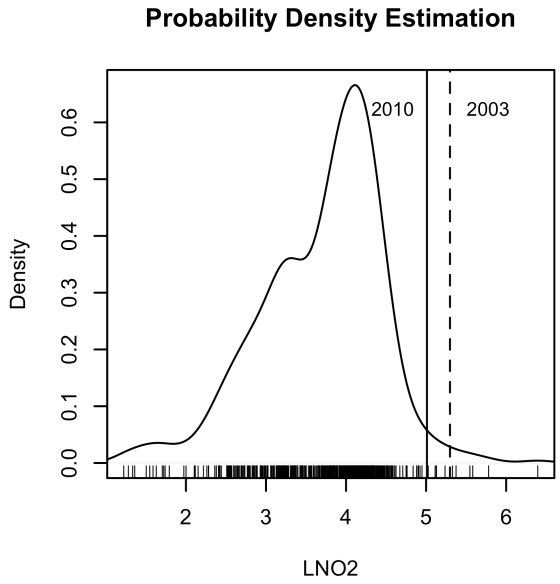


Figure 3.1: The pdf of the log-transformed NO_2 hourly concentration values (`LNO2`).

lower concentration values. Based on this information, predictions of these values should have associated costs/benefits assigned accordingly. In the following sections, we will explain why this makes standard evaluation approaches unsuitable for this kind of applications.

3.2.2 The Inadequacy of Standard Error Measures

The preference criteria most commonly used in regression are the Mean Squared Error (MSE) and the Mean Absolute Deviation (MAD), defined as follows,

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2\,, \tag{3.4}$$

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.5}$$

Standard error measures, such as these ones, are not suitable in the context of a regression problem with non-uniform costs. Their weakness lies in taking all the prediction errors equally across the domain of the target variable. They assume that the magnitude of the committed error is the decisive factor for the cost assigned to a prediction.

In order to confirm our claim, we created two artificial set of predictions for the prediction task of outdoor air pollution (Aldrin and Haff, 2005) of Example 3.1. Our objective with this example

is to illustrate that, whenever costs/benefits are involved, standard error measures are no longer effective, namely for the comparison/selection of regression models.

Example 3.1. Performance estimates with standard error metrics: prediction of NO_2 emissions.

The outdoor air pollution prediction task is one example of an application where the importance of the target variable values, i.e. `LNO2` *hourly concentration values, varies throughout its domain.*

For 10 observations of `LNO2`*, we have two artificial sets of predictions made by two hypothetic models:* M_1 *and* M_2 *(cf. Table 3.2).*

Table 3.2: A new set of predictions can be produced from an original set of predictions by re-allocating its errors to different values. The overall error magnitude of the new set is the same of the original one.

Predictions of Two Artificial Models										
True	2.71	3.35	3.36	3.63	4.08	4.16	4.31	5.55	5.78	6.40
M_1	2.68	3.3	3.43	3.72	3.96	4.29	4.55	5.91	7.03	4.72
M_1 **Loss**	0.03	0.05	0.07	0.09	0.12	0.13	0.24	0.37	1.25	1.67
M_2	1.04	4.61	3.73	3.87	4.21	4.04	4.41	5.62	5.73	6.37
M_2 **Loss**	1.67	1.25	0.37	0.24	0.13	0.12	0.09	0.07	0.05	0.03

The scatter plot in Figure 3.2 shows these predictions and the true values.

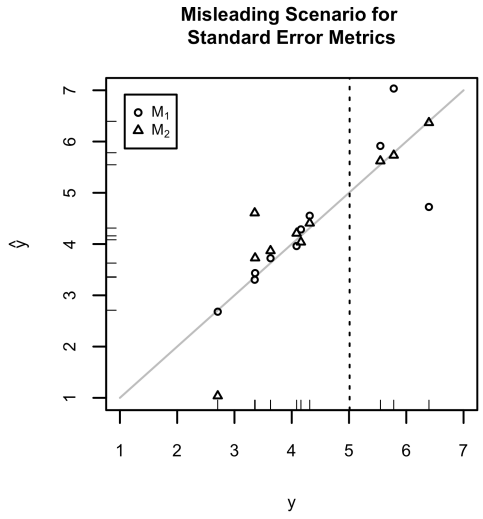


Figure 3.2: Two artificial sets of predictions for NO_2 emissions.

While M_1 *produces more accurate predictions at the lower* LNO2 *concentration values,* M_2 *achieves more accurate predictions at the higher* LNO2 *concentration values. Therefore, according to the application preference bias, one would say that* M_2 *is more useful than* M_1*.*

In order to get a more precise estimate of the performance of the two models, we evaluated both of them by MAD *and* MSE*. From the analysis of the obtained performance estimates, shown in Table 3.3, we notice that even though the predictions made by the two models are quite different, they are not distinguishable by any of the two standard error metrics. The reason is that both models sum up to the same total error value and thus both have the same average cost according to the standard metrics.*

Table 3.3: Two different prediction sets with the same standard error estimate.

Estimated Performance		
	MAD	**MSE**
M_1	0.402	0.460
M_2	0.402	0.460

In effect, M_1 *and* M_2 *are two hypothetical models with artificially generated predictions sets designed to illustrate the drawbacks of standard metrics on regression tasks for cost-sensitive applications.* M_2 *is obtained from* M_1 *in such a way that the smaller errors (in amplitude) are allocated to the cases which have high values on the target variable. This means that both models have exactly the same error amplitudes but these occur in different ranges of the target variable,* M_2 *having the smaller errors on the most important cases.*

The problem is that the properties that we have presented before concerning the notion of usefulness in standard regression metrics, no longer apply. For instance, with model M_1 *we get* $L(2.68, 2.71) = 0.03$ *and with model* M_2 *we get* $L(6.37, 6.40) = 0.03$*. Still, from the perspective of this application, the usefulness of the first prediction should be much lower (i.e.* $U(2.68, 2.71) \ll U(6.37, 6.40)$*). In effect, the second situation forecasts a high* NO_2 *concentration value that will probably trigger some alarm. On the contrary, the first prediction forecasts an average* NO_2 *concentration value and, in this sense, we can say that it is much less useful. This small example contradicts the property of equal accuracy predictions being equivalent as established in standard regression (page 66). Another example can be found for model* M_1 *where* $L(2.68, 2.71) < L(4.72, 6.40)$ *but, from the application perspective, we should have* $U(2.69, 2.71) < U(4.72, 6.40)$*. This case contradicts the property of higher accuracy predictions being always better as established in standard regression (page 66). In spite of being equally or even more accurate, the usefulness of the prediction depends*

on the range of values involved as higher `LNO2` *concentration values are the most important to predict accurately.*

For this type of applications, it cannot be assumed that the error amplitudes cost the same across all the domain of the target variable. All standard error metrics make that assumption and, as such are not very helpful. They may even lead to counter-intuitive model evaluation/comparison that can have serious impacts on the use of predictive models on this type of real world applications. Therefore, it is necessary to have an error metric that is sensitive to the location of the errors within the range of the target variable, i.e. that is able to cope with differentiated costs/benefits across the domain of this variable.

Similarly to what happens with accuracy in classification, and as referred in several studies (e.g. Domingos, 1999; Elkan, 2001), these standard error measures are not reliable for addressing applications with differentiated costs/benefits.

3.2.3 Alternative Approaches and their Limitations

To overcome some of the drawbacks we have mentioned, several alternatives have been considered. We present some of them here and briefly explain their limitations.

3.2.3.1 A Classification Approach

Classification is currently the most commonly used approach to address a cost-sensitive problem. The interest for real-world data mining tasks made the area of learning with costs widely studied and, therefore, many learning techniques have been developed within this research field. In this context, we could adopt one of the cost-sensitive learning techniques (e.g. Elkan, 2001; Weiss and Provost, 2001) or one of the more recent utility-based learning techniques (e.g. Weiss et al., 2008) to address our target problems. This would be a straightforward solution. In this thesis we question this simple approach. Regardless of the obtained results, which will be discussed in Chapter 6, the application of these classification techniques requires a change in the problem definition (e.g. Torgo and Gama, 1997). Namely, the continuous target variable needs to be discretized into a set of bins and adequate misclassification costs need to be found to reflect the implicit ordering among these bins. This implies the definition of the number of bins and the respective cut points which is dependent on domain knowledge. We argue that in regression problems such as the ones we have

been describing, this process results in creating crisp and artificial divisions between the values of the target variable. The first consequences of such process are illustrated in the Example 3.2.

Example 3.2. Performance estimates with misclassification costs: prediction of NO_2 emissions

---

Suppose that we decide to divide the `LNO2` *concentration values in two classes, Clean Air (*`clean`*) and Polluted Air (*`polluted`*) according to the 2010 objective established in the Directive 2008/50/EC (cf. Equation 3.6) and define the respective cost matrix afterwards (cf. Table 3.4).*

$$class(\texttt{LNO2}) = \begin{cases} \texttt{clean}, & \textit{if } \texttt{LNO2} \leq \ln(150\mu/m^3); \\ \texttt{polluted}, & \textit{if } \texttt{LNO2} > \ln(150\mu/m^3). \end{cases} \tag{3.6}$$

Table 3.4: A cost matrix for the prediction of NO_2 emissions.

Cost Matrix			
		True	
		`clean`	`polluted`
Predicted	`clean`	$c_{\texttt{c,c}}$	$c_{\texttt{c,p}}$
	`polluted`	$c_{\texttt{p,c}}$	$c_{\texttt{p,p}}$

According to some researchers (e.g. Provost et al., 1998), the task of assigning costs is not trivial. Nevertheless, given the objectives of this concrete application, the costs associated to errors committed with the class `polluted` *should be more important than the rest of the errors. This means that* $c_{\texttt{p,p}}$*,* $c_{\texttt{p,c}}$ *and* $c_{\texttt{c,p}}$ *should be carefully chosen.*

A prediction of $\ln(149\mu/m^3)$ *for a true concentration value of* $\ln(150\mu/m^3)$ *would incur on a cost of* $c_{\texttt{c,p}}$*. Meanwhile, a prediction of* $\ln(200\mu/m^3)$ *would be considered as accurate, achieving the same benefit of a perfect prediction of* $\ln(150\mu/m^3)$*:* $c_{\texttt{p,p}}$*. These two examples are clearly counter-intuitive. The problem is that, according to the cost matrix, all the dangerous high concentrations of* `LNO2` *are treated equally.*

In order to make the evaluation more sensitive to variations within high concentrations of `LNO2`*, we could divide the range of values of the target variable further. We could create one or more extra classes within the high and dangerous concentrations values of* `LNO2`*, thus creating a kind of degree of pollution. Nevertheless, we still have to choose cut points to define the bins. Those threshold values will always be critical as they create crisp and artificial divisions between the values of a continuous variable.*

Moreover, in most of these applications, the most interesting class, where prediction errors are more costly, is also the less frequent one. This is also the case of the `LNO2` *concentration values prediction. Fortunately, the number of times in which the concentration of* NO_2 *goes beyond the imposed thresholds is low. This means that the class* `polluted` *is not frequent and thus splitting it in more bins would create a set of classes that are even less frequent. As mentioned by several authors (e.g. Weiss, 2004) these type of class unbalance is very hard to address.*

From our perspective, this classification approach is counter-intuitive for addressing regression tasks. A 0/1 loss function is not suitable for problems that are essentially metric, such as the ones we want to address. We claim that this is a rough approximation of a problem that is inherently continuous. Still, since there are much more studies concerning the use of the notion of utility in classification (e.g. Weiss et al., 2005, 2008; Zadrozny et al., 2006), we will include this approach in our experimental comparisons, presented in Chapter 6.

3.2.3.2 High Powers on Error Differences

The error estimates provided by the Mean Squared Error (MSE) statistic (cf. Equation 3.4 - page 69) are not robust. This means that the performance estimates given by this metric are influenced by the presence of extreme high or low values in the data. As it is based on the squared error, larger errors, usually committed at extreme values, have more weight on the error estimates. In comparison, the Mean Absolute Deviation (MAD) (cf. Equation 3.5 - page 69) is less influenced by the presence of such extreme values as it is based on the absolute error. Therefore, the MAD estimates are more stable with respect to small changes in the data.

In this context, if we want our model to be sensitive to extreme values, then MSE would be a better choice than MAD for evaluation criterion. Moreover, if we use higher powers on the error magnitudes, as shown in Equation 3.7, that would make the performance on the extremes even more important in the overall performance of the model. This means that the smaller the error estimate obtained by this error statistic is, the more accurate the model at the extreme values probably is.

$$Err^k = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|^k \tag{3.7}$$

where $k \in \mathbb{N}$.

Nevertheless, this approach suffers from the same weakness as MAD and MSE: it has no sensitivity to where the error occurs in the range of the target variable Y and no extra weight is given to the performance on extreme values. Thus, the same problems can arise, no matter which value of k [*4] is chosen.

3.2.3.3 Case Weights

In the context of regression approaches we can use case weights as shown in Equation 3.8, to address our target applications.

$$Err_w = \frac{\sum_{i=1}^{n} w_i \cdot L(y_i, \hat{y}_i)}{\sum_{i=1}^{n} w_i} \tag{3.8}$$

where $L(y_i, \hat{y}_i)$ is a loss function and w_i is the weight associated to the case i.

The objective is that higher weights are given to the more important cases so that the model parameters can be obtained by minimizing a criterion that takes into account these weights.

In the Example 3.3, we have applied this approach to the prediction of outdoor pollution.

Example 3.3. Performance estimates with a weighted error metric: prediction of NO_2 emissions.

Let us consider the NO_2 concentration data set once more. The predictions set of M_1 and M_2, shown in Figure 3.3, were obtained by the same process explained in Example 3.1 (page 70). Here we are only considering their predictions on two test cases. M_2 is artificially obtained from M_1 by just exchanging the errors of the two predictions. M_2 gets the smaller error at the higher `LNO2` *concentration value.*

If we attach appropriate case weights to the errors, Err_w should be able to select M_2 as the best model. Nevertheless, and even though M_2 does the most accurate prediction at the most important case, this would only partially meet our application objectives.

By assigning higher weights to the higher `LNO2` *concentration values, the prediction of a low* `LNO2` *concentration value for a true high* `LNO2` *concentration value is penalized, and, thus the cost of predictions such as p_1 (see Figure 3.3) is increased. Within the context of this application, the*

[*4] When $k = 2$, Err^k is equivalent to the Mean Squared Error (MSE).

prediction p_1 *can be regarded as a* missed event *with a certain* opportunity cost*, as there was an occurrence of a high concentration of* LNO2 *which was missed, i.e. not predicted by model* M_1*.*

While missed events are correctly considered by this approach, the same does not occur with the following situation. Suppose we have the prediction of a high LNO2 *concentration value for a true low* LNO2 *concentration value. This is the case of* p_2 *(see Figure 3.3), which is assigned a low weight as these are solely based on the true value. Predictions such as* p_2 *are known as* false alarms*.*

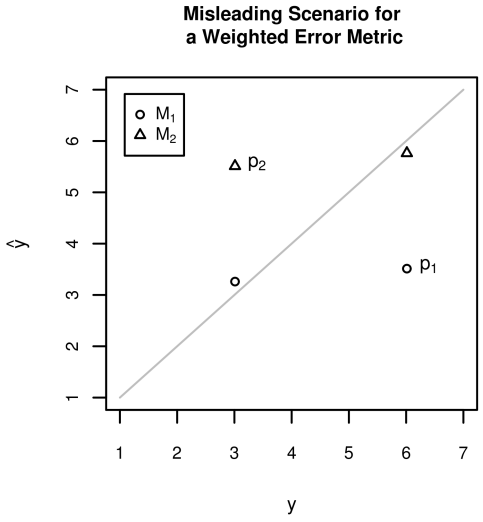


Figure 3.3: Two different models with a possible misleading Err_w estimate.

Even though we can assume that proper case weights could be easily obtained according to the target application, it is important to notice that when asserting the cost of a prediction, it is necessary to take into account both the true and the predicted values. Predicting an insignificant value for a test case that has a critical value is not the only cost we can incur. In effect, it may be equally serious to predict a critical value for a test case that has an insignificant value on the target, as we would be issuing a kind of *false alarm* that could also lead to serious costs.

3.2.3.4 Asymmetric Loss Functions

Christoffersen and Diebold (1996) and Crone et al. (2005) have addressed the issue of differentiated prediction costs in the context of financial applications, proposing asymmetric loss functions. Their main goal was to be able to distinguish between two types of errors and assign costs accordingly, namely, the costs of under-predictions ($\hat{y} < y$) and over-predictions ($\hat{y} > y$). In this context, they proposed the $LINLIN$ cost function, a new error metric defined as follows,

$$LINLIN = \begin{cases} c_o|y - \hat{y}|, & \text{if } \hat{y} > y; \\ 0, & \text{if } \hat{y} = y; \\ c_u|y - \hat{y}|, & \text{if } \hat{y} < y. \end{cases} \tag{3.9}$$

where c_o and c_u are constants to penalize over and under predictions.

The $LINLIN$ metric allows to differentiate the cost of the errors depending on where they occur, namely an error with certain amplitude may be more penalized than another error with the same amplitude if both occur on different "sides" of the true value, provided the constants c_o and c_u are different.

Example 3.4 shows how an asymmetric evaluation metric such as $LINLIN$, would apply to our target applications.

Example 3.4. Performance estimates with an asymmetric loss metric: prediction of NO_2 emissions.

Using the NO_2 *concentration data set, we have created other two artificial models,* M_1 *and* M_2*. Each of the models made two predictions: one is an over-prediction and the other is an under-prediction, as shown in Figure 3.4.*

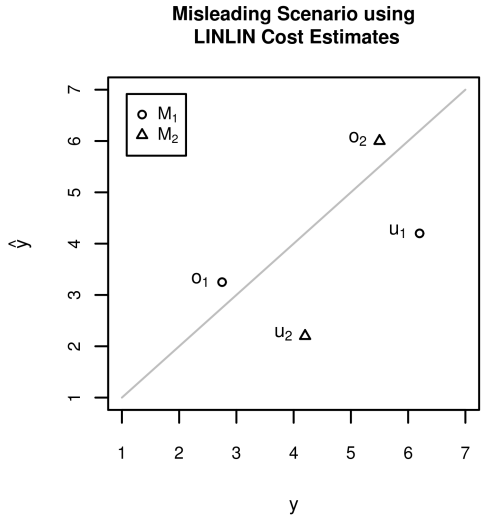


Figure 3.4: Two different models with the same $LINLIN$ cost.

The magnitude of the errors associated to the two under-predictions is equal, as well as the magnitude of the errors associated with the two over-predictions. This implies that, according

to the $LINLIN$ error metric, and whatever are the c_o and c_u defined, the two models are ranked ex-aequo.

However, though u_1 *and* u_2 *are both under-predictions, they occur at different ranges of the target variable. Thus, consistently with the application preference bias,* u_1 *should be associated to a higher cost then* u_2*. Similarly,* o_2 *should be more penalized then* o_1*. This is not possible in this framework and thus the evaluation provided by $LINLIN$ fails regarding the objectives of our target applications.*

Although this is a step in the direction of cost-sensitive regression, this error metric is only meant to distinguish two types of errors, depending on their signals. Moreover, it considers all under- (over-) predictions as equally serious, taking only into consideration the error magnitude as in the standard error metrics. As such, this approach is far from satisfying our goal of having a general methodology for handling cost-sensitive regression tasks.

The confinement of this cost error function can be specially penalizing in domains such as stock market forecasting. Regarding this setting, predicting a future price change of -1% for a true value of 1%, has the same error amplitude as predicting 6% for a true value of 8%. The two cases are under-predictions, and thus would have the same cost using the $LINLIN$ metric. However, these two situations are completely different from an investor's perspective. While the first prediction could lead to a wrong sell action that would in effect result in a loss of money, the latter would lead to a correct and profitable action. These problems arise because all these measures look at the error amplitudes independently of where they occur in the range of the target variable. As mentioned in Torgo (2005), it is important to study the performance of the models as a function of the target variable range for this type of applications.

Therefore, in any regression problem where the application preference bias is not uniform across the domain of the continuous target variable, a careful inspection of the evaluation metrics used in the selection/comparison of models should be made. We must check if their evaluation really reflects the appropriate preference bias for the application. With a set of small examples we have illustrated the inadequacy of the existing error metrics. Because they are insensitive to the concrete values involved in the predictions and, mostly, focus on the error amplitudes, they give some counter-intuitive indications concerning the comparison of models.

3.3 Utility in Regression

As mentioned in Elkan (2001) and Zadrozny (2005), research on cost-sensitive learning has traditionally been formalized in terms of costs as opposed to benefits or rewards. However, evaluating a model in terms of benefits is generally preferable because there is a natural baseline from which to measure all benefits whether positive (as real benefits of predictions) or negative (as costs of the predictions). Some studies (e.g. Daskalaki et al., 2006; Weiss et al., 2008) also refer that performance measures should adopt the 'business' objectives, which are usually proposed by the user. According to these studies, driving the data-mining process by these objectives is determinant to achieve potentially useful results. However, there is a lack of publicly available data where costs or utility objectives are precisely indicated.

Our proposal consists on the evaluation of regression models through an utility surface. This surface can be interpreted as a continuous version of the benefit matrix proposed by Elkan (2001). In order to obtain such surface, we need to calculate the utility of each prediction and a key notion is introduced: the *relevance* of a value in the context of the target application. The *relevance* is the crucial property that distinguishes non-uniform benefit/cost regression problems from standard regression problems. The fact is that, for our target applications, not all values of the target variable are equally relevant for the user. There may exist a range of so-called relevant values where it is particularly useful to be accurate. In this context, the driving intuition behind our proposal is the following.

Definition 3.3 (**Principle of Utility in Non-Uniform Regression**)**.** Utility is a function of both the error of the prediction (i.e. $L(\hat{y}, y)$) and the relevance (importance) of both $\hat{y}$ and y.

Together, the relevance and loss information on the predictions gives us an utility surface, which is then used to obtain an evaluation score that provides a more reliable estimate of a regression model effectiveness regarding the goals of these applications.

3.3.1 The Relevance of the Target Variable Values

We propose to model the relevance given by the user to the different values of the target variable by means of a relevance function. This function is domain-dependent and maps each value of the target variable into a scale of relevance. For some type of cost-sensitive classification problems, specifying the relevance of the target variable is essentially choosing the positive class. In regression, given the infinite nature of the domain of the target variable, a continuous real valued function makes more sense.

Definition 3.4 (**Relevance Function**)**.** The relevance function $\phi(Y) : \mathcal{Y} \rightarrow [0,1]$ is a continuous function that expresses the application-specific bias concerning the target variable domain $\mathcal{Y}$ by mapping it into a $[0,1]$ scale of relevance, where 0 represents the minimum and 1 represents the maximum.

Additionally, based on the relevance function ϕ, we define another function that, for any prediction, gives us the joint relevance value associated to it, as follows.

Definition 3.5 (**Joint Relevance Function**)**.** Let ϕ denote the relevance function $\phi\colon \mathcal{Y} \rightarrow [0,1]$ and $p \in [0,1]$ a weight parameter. We define the joint relevance of a given prediction by the function $\phi^p : \mathcal{Y} \times \mathcal{Y} \rightarrow [0,1]$ defined as the weighted mean of their relevance values, as follows

$$\phi^p(\hat{y}, y) \quad := \quad (1-p)\,\phi(\hat{y}) \;+\; p\,\phi(y) \tag{3.10}$$

where y and $\hat{y}$ are the true and predicted values.

As the objective of the relevance function is to represent domain knowledge, we do not impose any particular shape to the ϕ function. We assume that the specification is provided by the end-user. Still, specifying such function in an analytical way may not always be easy. It is also virtually impossible to describe reasonable default relevance functions for all non-uniform utility applications. As we will see later on, the relevance function can be defined through existing functions and techniques, which, for these type of applications, provide reasonable approximations to the applications objectives. Nevertheless, what is important to remark at this point is that, unlike standard regression, the values of the target variable are not equally relevant in non-uniform utility. There are regions of the target variable more relevant than others and this leads to the notion of *bumps of relevance*.

3.3.1.1 Bumps of Relevance

While in standard regression problems all values in the domain of the target variable are equally important and thus have constant relevance, in non-uniform utility regression we have *bumps of relevance*. Using bumps, the end-user emphasizes the relevance of some specific ranges of the target variable. At these ranges, the relevance function exhibits the shape of a bump. Bumps correspond, in fact, to intervals of the target variable where the relevance function is quasiconcave.

Quasiconcave functions are widely used in many research fields (e.g. microeconomics) due to some of their interesting properties. Quasiconcave functions are characterized as *single-peaked* functions like the probability density function of distributions such as uniform, normal, exponential, logistic,

Weibull, Gamma, among other distributions. In microeconomics (Friedman and Sandow, 2010; Guerraggio and Molho, 2004; Hendrix and Toth, 2010), for instance, these type of functions are called utility functions and measure an investor's relative preference for different levels of total wealth. As the objective is to maximize the utility based on the wealth, knowing in advance the shape of the function constitutes an advantage. Nevertheless, we should stress that our goal is to address applications where the relevance of the target variable may not be uniform, and, in particular, may not be *single-peaked*. Still, in many applications, relevance is often associated with specific ranges or with extremeness and rarity (e.g. highly profitable customers; high variations on stock prices; extreme weather conditions). Our proposal is to locally address these ranges as a way to achieve the global applications' objectives. Our ultimate goal is to maximize utility, and that can only be achieved by, simultaneously, maximizing the relevance and minimizing the error. Some of the properties of quasiconcave functions are at the partial fulfillment of this goal and, as such, we present them next.

Having defined the relevance function $\phi\colon \mathcal{Y} \to [0,1]$, a bump is an interval of the domain, $\mathcal{B} \subseteq \mathcal{Y}$, where the function ϕ is quasiconcave. In this context, we can equivalently say that:

(i) $\forall y_1, y_2 \in \mathcal{B}\ \forall \lambda \in [0,1]\ :\ \phi(\lambda\, y_1 + (1-\lambda)\, y_2) \geq \min\{\phi(y_1), \phi(y_2)\}$

(ii) the *upper contour set* $\mathrm{UC}_\phi(\mathcal{B}, \alpha) := \{y \in \mathcal{B}\colon \phi(y) \geq \alpha, \alpha \in \mathbb{R}\}$ is convex, i.e.
$\forall y_1, y_2 \in \mathrm{UC}_\phi(\mathcal{B}, \alpha)\ \forall \lambda \in [0,1]\ :\ \lambda\, y_1 + (1-\lambda)\, y_2 \in \mathrm{UC}_\phi(\mathcal{B}, \alpha)$.

Figure 3.5 presents an arbitrary relevance function defined in the context of non-uniform utility regression. Given our bump concept, this relevance function has four quite different bumps. Each one respects an interval where the function ϕ fulfills the conditions of quasiconcavity mentioned above.

Roughly speaking, quasiconcave functions are generalized concave functions, which are either monotonic, or first non-decreasing and then non-increasing. In effect, in spite of their restrictions, quasiconcave functions can assume several forms and, thus, easily adapt to the application goals. Besides their flexibility, quasiconcave functions have some interesting properties, which have made them useful in many contexts (Hendrix and Toth, 2010), such as microeconomics, as we have referred. Namely, the Local-Global Property of the Maximum, enunciated by Theorem 3.7, which in conjunction with Theorem 3.6, ensures the existence of an unique maximum.

Theorem 3.6. *Let $f : X \subset \mathbb{R} \to \mathbb{R}$ be a concave function, then it is also quasiconcave.*

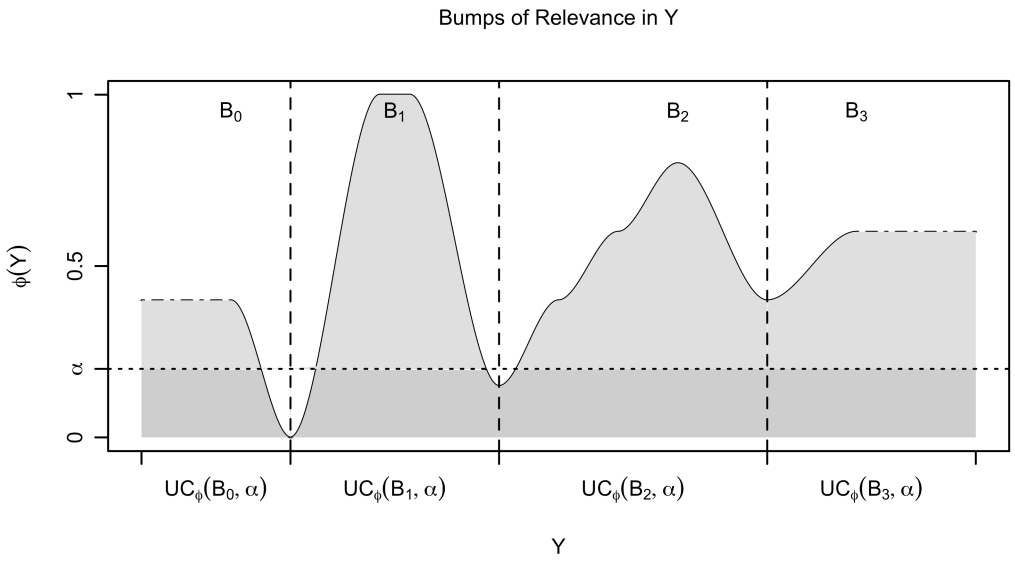


Figure 3.5: At the bumps, the relevance function ϕ is quasiconcave. This means that, at each bump, the upper contour set UC_ϕ is convex.

.

Proof. The concavity of f implies that, for all $x_1, x_2 \in X$ and for all $0 \leq \lambda \leq 1$,

$$\begin{aligned} f(\lambda\, x_1 \;+\; (1-\lambda)\, x_2) \;&\geq\; \lambda\, f(x_1) \;+\; (1-\lambda)\, f(x_2) \\ &\geq\; \lambda \min\{f(x_1), f(x_2)\} + (1-\lambda) \min\{f(x_1), f(x_2)\} \\ &\geq\; \min\{f(x_1), f(x_2)\} \end{aligned}$$

So f is quasiconcave. □

Theorem 3.7 (Local-Global Property of the Maximum)**.** *Let $f : X \subset \mathbb{R} \to \mathbb{R}$ be a concave function. Then*

(a) every local maximum of f is a global maximum,

(b) the set of maximizers of f on X, $\operatorname{argmax}_{x \in X} f(x)$*, is either empty or convex.*

Proof. (a) Let x_0 be any local maximum of f , but not a global maximum on X. Since x_0 is not a global maximum, then there is a subset $X_0 \subset X$, such that $f(x) \leq f(x_0)$, $x \in X_0 \cap X$. As x_0 is not a global maximum, a point $x_1 \in X$ exist such that $f(x_1) > f(x_0)$. We know that any point in X can be written as a linear combination of these two values. Hence, for any $x \in X$, there is

$\lambda \in [0,1]$, such that $x = (1-\lambda)\, x_0 \ + \ \lambda\, x_1$. The concavity of f implies

$$\begin{aligned} f(x) &= f((1-\lambda)\, x_0 \ + \ \lambda\, x_1) \\ &\geq (1-\lambda)\, f(x_0) \ + \ \lambda\, f(x_1) \\ &> (1-\lambda)\, f(x_0) \ + \ \lambda\, f(x_0) \\ &= f(x_0) \end{aligned}$$

So the concavity of f and the assumption $f(x_1) > f(x_0)$ implies that all points between x_1 and x_0 have an objective value higher than $f(x_0)$.
But, if we chose λ small, the point x is situated in X_0 and by hypothesis, x_0 was a local maximum in this interval, which is a contradiction. So the assumption that a point x_1 exists with $f(x_1) < f(x_0)$ cannot be true.

(b) Suppose that x_1, x_2 are both maximizers of f on X, i.e. for any $x \in X$, $f(x) \leq f(x_1)$ and $f(x_1) = f(x_2)$. For any $x \in X$, there is $\lambda \in [0,1]$, such that $x = (1-\lambda)\, x_1 \ + \ \lambda\, x_2$. The concavity of f, implies that

$$\begin{aligned} f(x) &= f((1-\lambda)\, x_1 \ + \ \lambda\, x_2) \\ &\geq (1-\lambda)\, f(x_1) \ + \ \lambda\, f(x_2) \\ &= (1-\lambda)\, f(x_1) \ + \ \lambda\, f(x_1) \\ &= f(x_1). \end{aligned}$$

The above must hold with equality or x_1 and x_2 would not be maximizers. Thus the set of maximizers is convex. □

Theorem 3.8. *A concave function $f : X \subset \mathbb{R} \to \mathbb{R}$ that attains a minimum over X, attains the minimum at some extreme point of X.*

Proof. Let x_0 be a minimizer of f in X, but not an extreme point of X. This means that there are two extreme points x_1, x_2 in X, such that $f(x_0) < f(x_1)$ and $f(x_0) < f(x_2)$. Additionally, x_0 can be written as a linear combination of x_1, x_2. Hence, there is $\lambda \in [0,1]$, such that $x_0 = \lambda\, x_1 \ + \ (1-\lambda)\, x_2$. By the concavity of f, we have

$$\begin{aligned} f(x_0) &= f(\lambda\, x_1 \ + \ (1-\lambda)\, x_2) \\ &\geq \lambda\, f(x1) \ + \ (1-\lambda)\, f(x2) \\ &> \lambda\, f(x_0) \ + \ (1-\lambda)\, f(x_0) \\ &= f(x_0) \end{aligned}$$

which is a contradiction. □

Based on the above theorems, the following corollary is drawn.

Corollary 3.9. *Let $f : X \subset \mathbb{R} \to \mathbb{R}$ be a quasiconcave function and x^* the global maximum of f in X. Then* $\inf(X)$ *is a global minimum of f for $\{x \in X : x \leq x^*\}$;*

Provided that some properties are proven for any quasiconcave function, we can apply them to bump regions defined by the relevance function. In this sense, the above results allow us to give a more rigorous definition of bump of relevance.

Definition 3.10 (**Bump**)**.** Let $\mathcal{Y} \subseteq \mathbb{R}$ be the domain of a continuous target variable and ϕ be the continuous relevance function associated to Y. A bump $\mathcal{B}$ is an interval of $\mathcal{Y}$ defined by the pair $\mathcal{B} := \langle b^-, b^* \rangle$ with $b^-, b^* \in \mathcal{Y}$ and $b^- \leq b^*$, where

(i) $b^* = \overline{\mathcal{B}_M}$, where $\mathcal{B}_M = \left\{ y \in \mathcal{B} \,|\, \operatorname{argmax}_y \phi(y) \right\}$

(ii) $b^- = \overline{\mathcal{B}_m}$, where $\mathcal{B}_m = \left\{ y \in \mathcal{B} \,|\, \operatorname{argmin}_{y \leq b^*} \phi(y) \right\}$

(iii) ϕ is quasiconcave, hence we have $\phi(b^-) \leq \phi(x) \leq \phi(b^*)$, where $x = \lambda\, b^- + (1 - \lambda)\, b^*$, $0 < \lambda < 1$;

More informally, each bump is defined by: (i) the average value b^- where the target variable attains the minimum relevance, before it grows to its maximum; (ii) the average value b^* where the target variable reaches the maximum relevance of the bump.

Definition 3.11 (**Bumps Partition**)**.** The set of all bumps

$$\mathcal{P}_\phi(\mathcal{Y}) := \left\{ \langle b_0^-, b_0^* \rangle, \langle b_1^-, b_1^* \rangle, \ldots, \langle b_r^-, b_r^* \rangle, \langle b_{r+1}^-, b_{r+1}^* \rangle \right\}$$

is called a bumps partition of $\mathcal{Y}$ with respect to the relevance function ϕ. In these conditions,

(i) $-\infty = b_0^- < b_1^- < \cdots < b_r^- < b_{r+1}^- = +\infty$ define the partition points;

(ii) each partition interval defines a bump, i.e. for each $0 \leq i \leq r$, the function $\phi : [b_i^-, b_{i+1}^-[\to [0, 1]$ is quasiconcave.

Figure 3.6 shows the bump partition obtained for the relevance function in Figure 3.5.

Additionally, we define another bump related notion, which will be used further on, regarding the utility function.

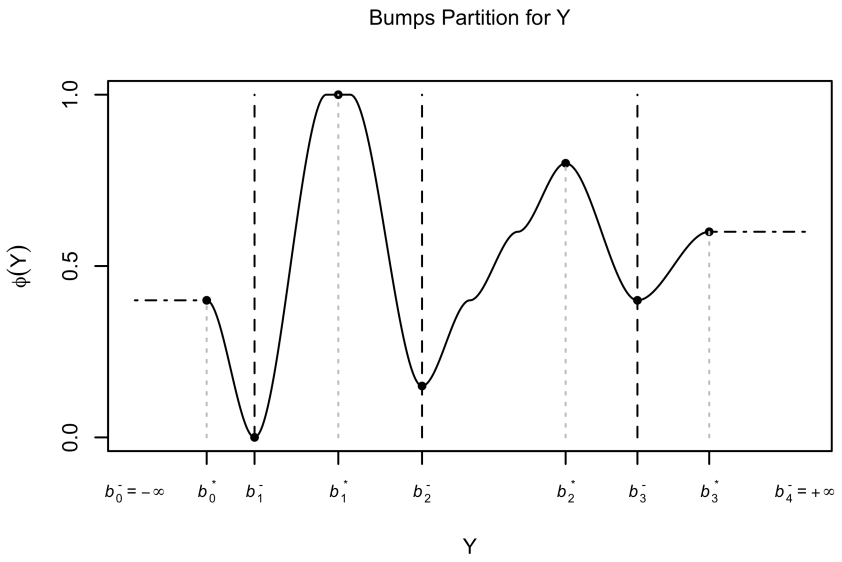


Figure 3.6: Bumps partition of Y with respect to relevance function ϕ. Each bump i is characterized by its partition node b_i^- and by one global maximum b_i^*.

Definition 3.12 (**Maximum Admissible Loss**)**.** The maximum admissible loss in a bump $\mathcal{B}_i$ is the double of the smallest amplitude in the bump, which is given by the difference between each one of its bounds, b_i^- and b_{i+1}^-, and its maximum value b_i^*, i.e.

$$b_i^\Delta := 2 \cdot \min\{|b_i^- - b_i^*|, |b_i^* - b_{i+1}^-|\} \tag{3.11}$$

The reasoning behind this definition is that this maximum admissible loss is a function of the minimum difference in terms of the target variable, as we move from the value with highest relevance in a bump (b_i^*) to a different bump. This means that for "narrow" bumps we are more sensitive to prediction errors, while for wider bumps we accept larger differences between the true and predicted values as reasonable.

The maximum admissible loss b_i^Δ is a finite value. Hence, if one of the two extreme bumps does not have a finite maximum admissible loss (e.g. $\mathcal{B}_0 = \langle -\infty, -\infty \rangle$), then it assumes the maximum admissible loss of its adjacent bump.

Taking the above definition, we indicate in Figure 3.7 the loss tolerance range defined for each bump of the relevance function shown in Figure 3.6.

3.3.2 From Prediction Errors to Utility Values

Our main motivation in the formulation of utility-based regression is to address applications where the target prediction variable has non-uniform relevance for the user. This non-uniform relevance

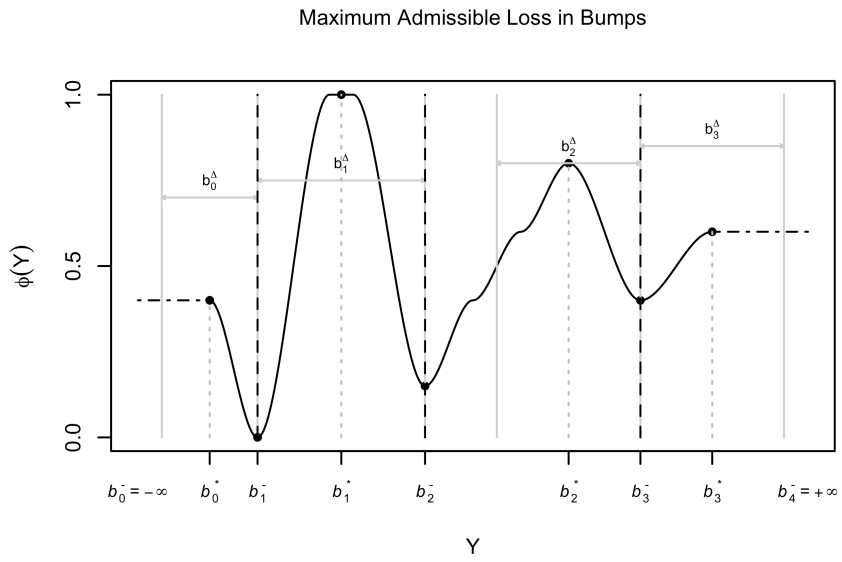


Figure 3.7: Maximum admissible loss in bumps: each bump has a maximum error tolerance defined by the double of the smallest amplitude in the bump between each of one of its bounds and its maximum value.

usually results from the fact that predictions may be actionable. In such domains, taking the right action results in positive benefits, while taking a wrong action results in costs (i.e. negative benefits) [⋆5]. As the set of possible actions is limited, this could lead to a typical classification setup. However, the other key distinction of our target applications is that we are interested in degrees of action and that is the key that takes us apart from classification approaches.

Two predicted values may eventually lead to the same action, but still one being preferred over the other. For instance, an investor in the stock market may be interested in selling but he may sell different quantities depending on the situation. A prediction of both 9% or 10% return for a true value of 11% may lead to the same action (which in this case would be a buy action as we are anticipating a high rise in prices). However, the 10% prediction should be considered more useful as it is clearly a more accurate anticipation of our profits. Moreover, a prediction of 10% may even lead to a larger investment than a prediction of 9%, which, given that the true raise end up being 11%, is even better.

In the context of utility-based regression, there are two aspects to consider for calculating the utility value of a prediction: (i) whether the predicted value leads to the correct action; and (ii) what is the precision of the predicted value. As we will see in the following sections, each prediction may entail some benefits and eventually also some costs. It is the balance between these two values that will result in the utility score of the prediction. The following sections provide the details on how we assess the benefits and the costs of a prediction.

[⋆5] From here forth, we will refer to *positive benefits* as benefits and *negative benefits* as costs.

3.3.2.1 Assessing Benefits

In the context of utility-based regression where predictions are actionable, making a perfect prediction means that we will be able to carry out the correct action. The concrete value of the benefit of this action should be proportional to the importance the user gives to the situation under consideration, i.e. the relevance of the true value $\phi(y)$. This means that a perfect prediction should achieve a benefit of $\phi(y)$, or at least something that is proportional to this quantity. This benefit should decrease as our predictions depart from the true value y. As our predictions deviate from the true value, there are two important criteria that determine if they can still be considered beneficial: (1) the predicted value still leads to the correct action, which within our framework means that the predicted value belongs to the same bump as the true value; and (2) the predicted value is "reasonably" accurate, which is related to the maximum admissible loss of the bump that we have defined before. If any of these conditions is not met, the prediction should have zero benefit, and, in effect should have a cost, as we will see later on.

Figure 3.8 provides an illustrative example of these two important criteria. Given a true value y and its bump index $\gamma(y)$ [*6], we show two possible predictions for this value, $\hat{y}_1$ and $\hat{y}_2$. We will use these two predictions as a means to introduce the notions that are necessary to calculate the benefits of a prediction using the two criteria outlined above.

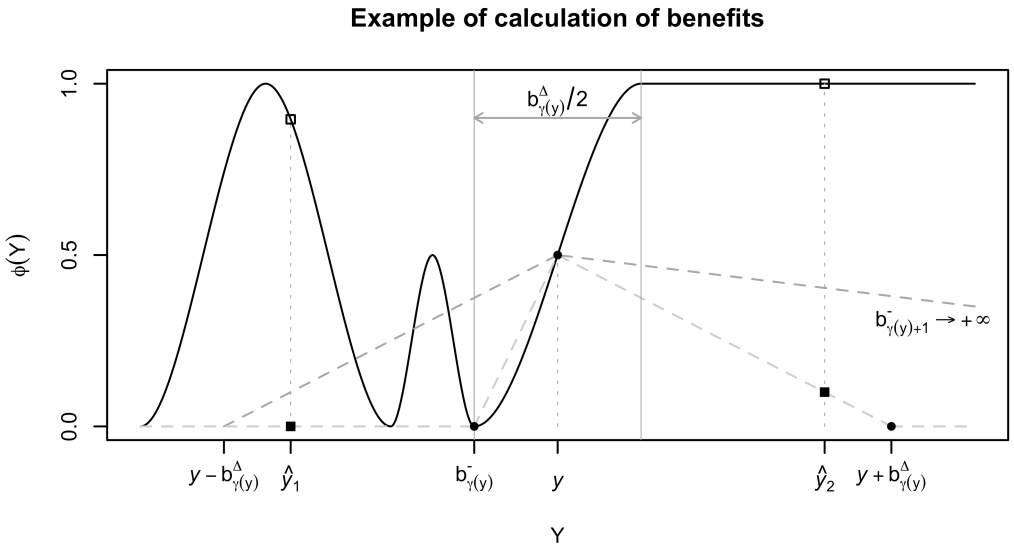


Figure 3.8: Illustration of the calculation of benefits for two different situations.

As we have mentioned the benefit of a prediction should be proportional to the importance the user gives to the true value. In this context, we propose the following definition of the benefit of a prediction:

[*6] $\gamma(y) := \{i \mid y \in \mathcal{B}_i \wedge \mathcal{B}_i \subseteq \mathcal{P}_\phi(\mathcal{Y})\}$

Definition 3.13 (**Benefit Function**)**.** The benefit of a prediction $\hat{y} \in \mathcal{Y}$ for a true value $y \in \mathcal{Y}$ is given by the following function:

$$B_\phi(\hat{y}, y) := \phi(y)\,(1 - \Gamma_B(\hat{y}, y)) \tag{3.12}$$

where ϕ is the relevance function and Γ_B is a bounded loss function.

The bounded loss Γ_B is a $[0, 1]$ function designed to assert the proportion of the maximum benefit a prediction should get. It should be 0 if we have a perfect prediction thus leading to the maximum benefits in the current situation, i.e. $\phi(y)$. The function should increase up to 1 as we move away from a perfect prediction or if we cross the boundaries of the bump to which y belongs. Formally, we propose:

Definition 3.14 (**Bounded Loss**)**.** The bounded loss of a prediction $\hat{y} \in \mathcal{Y}$ for a true value $y \in \mathcal{Y}$ is given by the following function:

$$\Gamma_B(\hat{y}, y) \quad := \quad \begin{cases} L(\hat{y}, y)/\dot{L}_B(\hat{y}, y), & \text{if } L(\hat{y}, y) < \dot{L}_B(\hat{y}, y) \\ 1, & \text{if } L(\hat{y}, y) \geq \dot{L}_B(\hat{y}, y). \end{cases} \tag{3.13}$$

where L is a "standard" loss function (e.g. absolute deviation) and $\dot{L}_B$ is a benefit threshold function.

We should remark that this bounded loss function will also be used in the definition of the costs of predictions to be presented in the next section, although in that context a cost threshold function $\dot{L}_C(\hat{y}, y)$ will be used instead of the benefit threshold function $\dot{L}_B(\hat{y}, y)$.

The benefit threshold function determines when the predicted value stops leading to a benefit. As we have mentioned before this may occur due to two conditions: (i) overcoming the maximum admissible loss of the bump; or (ii) being on a different bump. The following function implements these criteria.

Definition 3.15 (**Benefit Threshold**)**.** The benefit threshold of a prediction $\hat{y} \in \mathcal{Y}$ for a true value $y \in \mathcal{Y}$ is given by the following function:

$$\dot{L}_B(\hat{y}, y) := \min\{b^{\Delta}_{\gamma(y)},\, \ddot{L}_B(\hat{y}, y)\} \tag{3.14}$$

where $b^{\Delta}_{\gamma(y)}$ is the maximum admissible loss established for the bump of y, i.e. the bump index $\gamma(y)$ (cf. Definition 3.12), and $\ddot{L}_B$ is defined as follows,

$$\ddot{L}_B(\hat{y}, y) \quad := \quad \begin{cases} |y - b^{-}_{\gamma(y)}|, & \text{if } \hat{y} < y \\ |y - b^{-}_{\gamma(y)+1}|, & \text{if } \hat{y} \geq y. \end{cases} \tag{3.15}$$

This definition accomplishes the two above mentioned necessary conditions for a prediction to be considered a benefit. The first term in the min function is the maximum admissible error amplitude within the bump of the true value, which ensures that the predicted value is "reasonably" accurate. The second term in the min function checks if the predicted value yields to the correct action, by establishing the distance to the edges of the bump to which the true value belongs as limits.

In the particular example of Figure 3.8, predicted values smaller than y have a lower tolerance for error as they quickly reach one of the edges of the bump to which y belongs. Predictions below this edge ($b^-_{\gamma(y)}$) provide indications for a different action, according to this application relevance function. As the maximum admissible loss for the bump of y ($b^{\Delta}_{\gamma(y)}$) is larger than this value, this means that the benefit threshold for predictions smaller than y is $b^-_{\gamma(y)}$. In this sense, the maximum allowable error to get benefits for predictions below y is $\dot{L}_B(\hat{y}_1, y) = |y - b^-_{\gamma(y)}|$. As such, $\hat{y}_1$ (as well as any prediction below $b^-_{\gamma(y)}$) will get no benefit at all.

For the values above y the scenario is different. All predictions above y are included in the same bump of y because $b^-_{\gamma(y)+1} \to +\infty$. Thus, the tolerance to get a benefit is guided by the estimated maximum allowable error of the bump, $\dot{L}_B(\hat{y}_2, y) = b^{\Delta}_{\gamma(y)}$. Even though $\hat{y}_2$ is in the same bump of y its benefit ends up close to zero, because $\hat{y}_2$ is already rather near to the limit for getting benefits $y + b^{\Delta}_{\gamma(y)}$. Moreover, any value above $y + b^{\Delta}_{\gamma(y)}$, although in the same bump as y, will have null benefits as it is considered too distant from the true value.

Regards the behaviour of the proposed definition of benefits (Definition 3.13) we have that when $L(\hat{y}, y) = 0$, $B_\phi(\hat{y}, y) = \phi(y)$. This property follows from the definition of the bounded loss used in the definition of benefits. Moreover, when the loss value increases, our benefits will reach zero as shown below.

Lemma 3.16. *For sufficient large values of $L(\hat{y}, y)$, we have $B_\phi(\hat{y}, y) = 0$.*

Proof. We want to prove that there exists an $t_L \in [0, 1]$, such that for all $L(\hat{y}, y) \geq t_L$ we have $B_\phi(\hat{y}, y) = 0$. If we assign $t_L = \dot{L}_B(\hat{y}, y)$, then by the boundary loss function definition, we have $\Gamma_B(\hat{y}, y) = 1$ and hence $1 - \Gamma_B(\hat{y}, y) = 0$. Therefore, $B_\phi(\hat{y}, y) = 0$. □

In summary, the proposed benefit function ensures that the benefit is maximum for a perfect prediction and that it smoothly reaches zero as the prediction either is too inaccurate or it reaches the boundaries of the bump to which y belongs (i.e. leads to an incorrect action).

3.3.2.2 Assessing Costs

Having defined the notion of the benefits of a prediction we now turn our attention to the costs. With the exception of cases where the predicted value equals the true value, all predictions will have some cost. The costs result either because (i) they entail the wrong action, or (ii) they are inaccurate. Within our proposed framework the cost of a prediction is proportional to the impact of the wrong action it entails (if that is the case), and also to the distance between the predicted and true values (i.e. the loss).

In this context of actionable predictions there are three different types of incorrect actions: (i) *false alarms* where the prediction leads the user to a relevant event/action when the true value is rather irrelevant (i.e. we act when we should not) ; (ii) *missed opportunities* where the model predicts an irrelevant value but the true value is highly relevant (i.e. we did not do anything when we should have acted); or (iii) *confusing events* where the prediction leads to a wrong action (i.e. we ought to act but we carry out the wrong action). The third scenario is the most serious type of mistake.

While the benefits of a prediction depend on the usefulness of its associated action (i.e. $\phi(y)$), costs depend not only on the action associated with the true value but also on the action of the predicted value. This difference also results from the fact that when we are talking about benefits we are assuming that the right action was forecasted, while with costs that is not necessarily the case. This means that costs should be proportional to the relevance of both the true and predicted values. The joint relevance function (cf. Definition 3.5) captures this notion by calculating a weighted average of these two factors.

The cost of a prediction should thus be a quantity (a negative utility score) that is proportional to this joint relevance score. In this context, we propose the following definition of the cost of a prediction.

Definition 3.17 (**Cost Function**)**.** The cost of a prediction $\hat{y} \in \mathcal{Y}$ for a true value $y \in \mathcal{Y}$ is given by:

$$C^{p}_{\phi}(\hat{y}, y) := -\phi^{p}(\hat{y}, y)\, \Gamma_{C}(\hat{y}, y) \tag{3.16}$$

where ϕ^{p} is the joint relevance function and Γ_{C} is the bounded loss (c.f. Definition 3.14) calculated using the cost threshold function given below.

The cost threshold function determines when the costs reach the maximum value. As with the benefit threshold function this may occur due to two conditions: (i) overcoming the maximum admissible loss of the bump; or (ii) predicting a value that has the maximum relevance of a different bump (i.e. a different action). The next function implements these conditions.

Definition 3.18 (**Cost Threshold**)**.** The cost threshold of a prediction $\hat{y} \in \mathcal{Y}$ for a true value $y \in \mathcal{Y}$, is given by the following function:

$$\dot{L}_C(\hat{y}, y) := \min\{b^{\Delta}_{\gamma(y)}, \ddot{L}_C(\hat{y}, y)\} \tag{3.17}$$

where $b^{\Delta}_{\gamma(y)}$ is the maximum admissible loss established for the bump of y (cf. Definition 3.12) and $\ddot{L}_C$ is given by,

$$\ddot{L}_C(\hat{y}, y) \quad := \quad \begin{cases} |y - b^*_{\gamma(y)-1}|, & \text{if } \hat{y} < y \\ |y - b^*_{\gamma(y)+1}|, & \text{if } \hat{y} \geq y. \end{cases} \tag{3.18}$$

The first term in the min function is the maximum admissible error amplitude of the bump of the true value. The second term in the min function checks if the predicted value has reached the maximum relevance value of a neighbouring bump.

In Figure 3.9 we re-analyze the example presented in Figure 3.8, this turn in terms of the costs of the predictions. Contrary to the benefits, the costs increase in proportion with the loss function. The true value y and the prediction $\hat{y}_1$ are located at different bumps of the relevance function, which makes $\hat{y}_1$ an example of a wrong action. For predicted values below y (as $\hat{y}_1$) the cost increases till the maximum cost that is first reached at the point with highest relevance on the left adjacent bump $\ddot{L}_C(\hat{y}_1, y) = |y - b^*_{\gamma(y)-1}|$. Any prediction equal or below $b^*_{\gamma(y)-1}$ gets this same maximum cost.

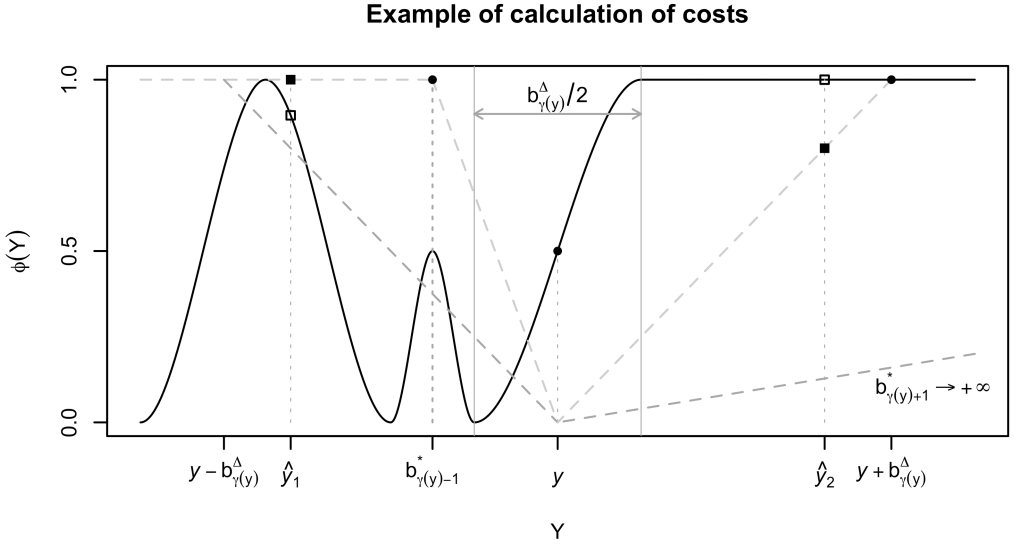


Figure 3.9: Illustration of the calculation of the costs for two different situations.

For predictions above y there is no adjacent bump to the right of the bump of y. This means that, as $b^*_{\gamma(y)+1} \to +\infty$, any prediction above y will lead to the same action as y. Still, after a certain

inaccuracy of the prediction the maximum cost is also reached. This means that predictions above y will reach this maximum if they are above $y + b^{\Delta}_{\gamma(y)}$. The cost of prediction $\hat{y}_2$ is near this maximum cost as it can be seen in Figure 3.9.

In terms of properties of the proposed definition of the cost of a prediction, we know that when $L(\hat{y}, y) = 0$, $C^p_\phi(\hat{y}, y) = 0$ because $ULC(\hat{y}, y) = 0$, but when the loss value increases our costs should be proportional to the relevance of the true and predicted value.

Lemma 3.19. *For sufficient large values of $L(\hat{y}, y)$, we have $C^p_\phi(\hat{y}, y) = -\phi^p(\hat{y}, y)$.*

Proof. We want to prove that there exists an $t_L \in [0, 1]$, such that for all $L(\hat{y}, y) \geq t_L$ we have $C^p_\phi(\hat{y}, y) = -\phi^p(\hat{y}, y)$. If we assign $t_L = \dot{L}_B(\hat{y}, y)$, then by the boundary loss function definition, we have $\Gamma_C(\hat{y}, y) = 1$. Therefore, $C^p_\phi(\hat{y}, y) = -\phi^p(\hat{y}, y)$ □

3.3.2.3 Final Utility Values

Having defined how to obtain the benefits and costs associated with any prediction we can now propose a method to calculate the utility of a prediction. Our proposal implements the following general principle:

The utility of a prediction is the net balance between its benefits and costs.

Definition 3.20 (**Utility Function**)**.** The **utility** of a prediction $\hat{y} \in \mathcal{Y}$ for a true value $y \in \mathcal{Y}$ is given by:

$$U^p_\phi(\hat{y}, y) := \phi(y)\,(1 - \Gamma_B(\hat{y}, y)) \; - \phi^p(\hat{y}, y)\,\Gamma_C(\hat{y}, y) \tag{3.19}$$

where ϕ^p is the joint relevance function with respect to the relevance function ϕ and the penalization factor p, Γ_B is the bounded loss function with respect to benefits threshold function $\dot{L}_B$, and Γ_C is the bounded loss function with respect to costs threshold function $\dot{L}_C$.

Theorem 3.21 (Limits of the Utility of a Prediction)**.** *Let $U^p_\phi \colon \mathcal{Y} \times \mathcal{Y} \to [-1, 1]$ be an utility function defined for a continuous target variable with domain $\mathcal{Y}$ and with a relevance function $\phi \colon \mathcal{Y} \to [0, 1]$. Then whatever is the predicted value $\hat{y} \in \mathcal{Y}$ for a true value $y \in \mathcal{Y}$, we have*

(a) the maximum of $U^p_\phi(\hat{y}, y)$ is $\phi(y)$;

(b) the minimum of $U^p_\phi(\hat{y}, y)$ is $p\,(1 - \phi(y)) - 1$.

Proof. (a) According to our utility function formulation, for the maximum value of utility to be achieved it is necessary that the prediction matches the true value, i.e. $\hat{y} = y$. This means that the loss value is zero, i.e. $L(\hat{y}, y) = 0$. From the definition of bounded loss function (cf. Definition 3.14) with respect to benefits threshold function $\dot{L}_B$, we have that if $L(\hat{y}, y) = 0$ then $\Gamma_B(\hat{y}, y) = 0$. Similarly, regarding bounded loss function with respect to cost threshold function $\dot{L}_C$, we have that if $L(\hat{y}, y) = 0$ then $\Gamma_C(\hat{y}, y) = 0$. Thus, in these conditions and considering that the relevance function ϕ only takes values in $[0, 1]$, it can be shown that,

$$\begin{aligned} U^p_\phi(\hat{y}, y) &\leq U^p_\phi(y, y) \\ &= \phi(y)\,(1 - \Gamma_B(y, y)) \; - \phi^p(y, y)\,\Gamma_C(y, y) \\ &= \phi(y) \end{aligned}$$

(b) From the zero loss prediction, the utility value decreases according to benefits and cost criteria that rely on the action that is implied by the prediction and on the loss value. In this sense, the worst scenario is achieved when the benefit is zero and the cost is maximum. Lemma 3.16 and Lemma 3.19 have established this scenario. When loss tends to infinity, i.e. $L(\hat{y}, y) \to +\infty$, we have $B_\phi(\hat{y}, y) = 0$ and $C^p_\phi(\hat{y}, y) = -\phi^p(\hat{y}, y)$. In these conditions and considering that both the relevance function ϕ and the penalizing cost factor p only take values in $[0, 1]$, it can be shown that,

$$\begin{aligned} U^p_\phi(\hat{y}, y) &= B_\phi(\hat{y}, y) + C^p_\phi(\hat{y}, y) \\ &= \phi(y)\,(1 - \Gamma_B(\hat{y}, y)) \; - \phi^p(\hat{y}, y)\,\Gamma_C(\hat{y}, y) \\ &\geq -\phi^p(\hat{y}, y) \\ &\geq p\,(1 - \phi(y)) - 1 \end{aligned}$$

□

3.3.3 Relationship with Standard Regression

In this section we explain how a standard regression problem can be recast as an utility-based regression problem.

Regarding our utility-based regression setup, the maximization of the utility implies the fulfillment of two criteria: the prediction of the correct action and the minimization of the prediction error. In a standard regression scenario, no actions are attached to the different values of target variable, i.e. all the values are considered equally relevant to the domain user. Thus, the minimization of

the prediction error is the only goal for standard regression. Still, we can describe such a scenario as an instance of our utility-based regression setup.

In effect, we can see standard regression as a problem where all values have equal and maximal relevance, i.e. $\phi(y) = 1, \forall y \in \mathbb{R}$. In this context, we have a single and infinite region of interest $\mathcal{Y} = (-\infty, +\infty)$, which corresponds to two theoretical bumps in the bumps partition, $\mathcal{P}_\phi(\mathcal{Y}) = \{\langle -\infty, -\infty \rangle, \langle +\infty, +\infty \rangle\}$.

Provided that the relevance function is constant, we have no indications on the benefits or loss threshold to define the utility loss functions Γ_B and Γ_C. In such conditions, we can assign the maximum admissible loss of the only existing bump with a standard error estimate ε of the target variable, such as the one proposed by Cherkassky and Ma (2004), based on the Gaussian level of noise and on the number of observations. This means that only the predictions with a loss below this estimated standard error ε are within the tube of benefit and cost analysis. Moreover, as we have mentioned, there are no actions attached to the target variable domain. Thus, the maximum admissible loss is the only criterion for a prediction to be considered as a benefit or a cost, and it is the same for both of them. In this context, based on this single error estimate, we can define a new utility loss threshold function as follows,

$$\Gamma_\varepsilon(\hat{y}, y) = \begin{cases} L(\hat{y}, y)/\varepsilon, & \text{if } L(\hat{y}, y) < \varepsilon \\ 1, & \text{if } L(\hat{y}, y) \geq \varepsilon. \end{cases} \tag{3.20}$$

where L is a loss function and ε the maximum admissible loss.

Considering the above setup, we have that if for $y \in \mathcal{Y}$, $\phi(y) = 1$ then, according to the definition 3.5 (page 80), it also true that for $\hat{y}, y \in \mathcal{Y}$, $\phi^p(\hat{y}, y) = 1$. Moreover, the existence of single error estimate ε, for both benefits and costs, makes the definitions of bounded loss functions Γ_B and Γ_C equivalent. In these conditions, and based on Equation 3.20, our utility function is reformulated as follows,

$$\begin{aligned} U^p_\phi(\hat{y}, y) &= \phi(y)\,(1 - \Gamma_B(\hat{y}, y)) - \phi^p(\hat{y}, y)\,\Gamma_C(\hat{y}, y) \\ &= 1 - \Gamma_B(\hat{y}, y) - \Gamma_C(\hat{y}, y) \\ &= 1 - \Gamma_\varepsilon(\hat{y}, y) - \Gamma_\varepsilon(\hat{y}, y) \\ &= 1 - 2\,\Gamma_\varepsilon(\hat{y}, y) \end{aligned}$$

With this utility function, a standard loss function L is scaled into a $[-1, 1]$ interval of utility values. The maximization of U is equivalent to the minimization of the bounded loss function Γ_ε, and thus the minimization of L, the goal of a standard regression task. Through this approach, any standard regression problem can be addressed in our utility-based regression framework.

3.3.4 Utility-based Performance Metrics

The function U_ϕ^p gives us a base to evaluate models for regression tasks with non-uniform benefits/costs across the target variable. Through it, we can estimate the expected utility of a model.

Definition 3.22 (**Expected Utility**). The **expected utility** of a regression model $\hat{f}_\Omega$ with respect to a set of instances $\langle \mathrm{x}, y \rangle \in \mathcal{X} \times \mathcal{Y}$ drawn from a distribution D is defined as

$$E_U(\hat{f}_\Omega, D) \;:=\; E_{\langle \mathrm{x},y\rangle \sim D}\,[\; U_\phi^p(\hat{f}_\Omega(\mathrm{x}), y)\;] \tag{3.21}$$

Based on the expected utility of a regression model, several performance measures can be derived, such as the Mean Utility (MU), shown in Equation 3.22.

$$\mathrm{MU} = \frac{1}{n}\sum_{i=1}^{n} U_\phi^p(\hat{y}_i, y_i) \tag{3.22}$$

where ϕ is the relevance function and p is the penalizing costs factor.

Using such metric we are able to estimate the performance of different models according to the preference biases of the applications (provided by relevance function ϕ and penalization factor p).

An equivalent metric is obtained if we map the values of MU to the interval $[0, 1]$, thus leading the Normalized Mean Utility NMU as follows,

$$\mathrm{NMU} = \frac{\sum_{i=1}^{n} U_\phi^p(\hat{y}_i, y_i) + n}{2\,n} \tag{3.23}$$

where ϕ is the relevance function and p is the penalizing costs factor.

3.4 Practical Implementation Issues of the Utility Function

Having defined the main principles of utility-based evaluation on a regression context, we now address some practical issues regarding the implementation of the utility function. We begin by illustrating how a relevance function can be drawn from a given set of known relevance points. We then explain how the bumps partition can be defined so that the utility function respects the application requirements. Additionally, we present a sigmoidal transformation of the utility function that can be applied if it fits the target problem better. Throughout this section, we revisit the Outdoor Air Pollution prediction task first presented in Section 3.2.1 (page 67).

3.4.1 Piecewise Cubic Hermite Interpolation of Relevance

As we have mentioned earlier in Section 3.3.1 (page 79), the information on the relevance of the target variable is application-specific and thus should be provided by the end-user. Still, specifying relevance through a continuous real-valued function, in order to map the domain of the target variable into a $[0, 1]$ interval scale as required, may not be straightforward for all applications.

Our proposal is to use *piecewise cubic Hermite interpolation* (Dougherty et al., 1989) to define the relevance function by interpolating a set of relevance values at the so-called *control points* specified by the end-user. Through interpolation it is possible to generate a function that goes through the set of n user-provided points. Splines are piecewise cubic interpolants that guarantee smoothness on the interpolation. Although splines are smooth, the lack of flexibility in their shape is limiting. For this reason, we chose the Hermite cubic polynomials as they allow more user control on the shape of the generated function, by requiring the derivative at the control points.

According to Dougherty et al. (1989), the piecewise cubic Hermite interpolating polynomials are simple and effective interpolants of discrete data. Moreover, by restricting the first derivative values at the *control points* they can preserve local positivity, monotonicity and convexity of the data. These are convenient properties in the context of our target applications as we want to induce a continuous function that reproduces, as closely as possible, the relevance points provided by the user.

Algorithm 3.1 shows how the piecewise cubic Hermite interpolation is performed over a set of given points S. One of the key points of this algorithm is to find the right slopes at the given points so that the interpolant is *piecewise monotone*, i.e. its derivative does not change sign in any interval defined by the control points. This task is ensured by check_slopes which implements a method proposed by Fritsch and Carlson (1980). This method estimates reasonable derivatives at each control point and ensures a zero derivative if the control point is a local maximum or minimum, so that the interpolation is locally monotone. Once the first derivative values are known, the four coefficients of the interpolant for each interval are then calculated.

To define the relevance function $\phi\colon \mathcal{Y} \to [0, 1]$, we should provide a set of control points $S = \{\langle y_k, \phi(y_k), \phi'(y_k)\rangle\}_{k=1}^{s}$ as input to the pchip algorithm. This set must contain information on the target value, its respective relevance value and on the derivative of the relevance function at that point. By default, we assume that the control points refer to local extrema of relevance and thus all have derivative $\phi'(y_k)$ equal to zero. Still, any other value can be provided. If necessary, check_slopes adjusts them so that the monotonicity is preserved. However, we do impose the derivatives to be zero at the two extreme control points, which necessarily makes them local

Algorithm 3.1 pchip(S): piecewise cubic Hermite interpolating polinomial.

Input: set of control points $S = \{\langle y_k, \varphi(y_k), \varphi'(y_k)\rangle\}_{k=1}^{s}$ with
$y_1 < y_2 < \cdots < y_s$, relevance values $\varphi(y_k)$ and preliminary slopes $\varphi'(y_k)$.
Output: $\phi(y)$: a piecewise cubic Hermite interpolating polynomial.

for $k \leftarrow 1$ **to** $s-1$ **do**
  $h_k \leftarrow y_{k+1} - y_k$
  $\delta_k \leftarrow (\varphi(y_{k+1}) - \varphi(y_k))/h_k$
  $a_k \leftarrow \varphi(y_k)$
end for
$\{b_k\}_{k=1}^{s-1} \leftarrow$ check_slopes($\{\varphi'(y_k)\}_{k=1}^{s-1}, \{\delta_k\}_{k=1}^{s-1}$) // Fritsch and Carlson (1980) method
for $k \leftarrow 1$ **to** $s-1$ **do**
  $c_k \leftarrow (3\delta_k - 2b_k + b_{k+1})/h_k$
  $d_k \leftarrow (b_k - 2\delta_k + b_{k+1})/h_k^2$
end for
return $\phi(y) = a_k + b_k\,(y - y_k) + c_k\,(y - y_k)^2 + d_k\,(y - y_k)^3,\ y \in [y_k, y_{k+1}]$

maxima or minima. As we proceed to a linear extrapolation, the relevance function is guaranteed to be constant outside the specified range of control points and thus, attached to the information provided by the end-user. At the end of the interpolation process, evaluating the relevance of a value $y \in \mathcal{Y}$, i.e. $\phi(y)$, corresponds to evaluate the interpolant polynomial $\phi(y)$ such that k is the interval of interpolation to which y belongs.

Example 3.5. Definition of relevance function by piecewise cubic Hermite interpolation: prediction of NO_2 emissions.

With respect to the NO_2 *Emissions prediction problem, we do not have an analytical definition of the relevance function* ϕ *from the domain. Still, based on the information provided by Directive 2008/50/EC and summarized in Table 3.1 (page 68), it is possible to gather the relevance at some key points. The aim of this application is to predict high concentration values of* NO_2 *Emissions. In particular, we know that the goal established for 2010 was to maintain the* LNO2 *hourly concentration values below a limit equal to* $ln(150\mu g/m^3)$*. This means that any concentration value that goes above this threshold is of major importance to the domain experts. In this context, we set the limit* $ln(150\mu g/m^3)$ *with maximum relevance. Using the same reasoning, we also set the alert value* $ln(400\mu g/m^3)$*, reported as a harmful limit, with maximum relevance. According to the Directive 2008/50/EC, the annual mean guideline for air quality is* $ln(40\mu g/m^3)$*. Based on this information, we decided to attach to this point an average relevance, such as 0.5. Finally, to complete our comprehension of the domain, we included the information that the lowest concentration value of* LNO2 *is irrelevant. Thus, we set the minimum* LNO2 *value that we have found on data, i.e.* $ln(3\mu g/m^3)$*, with minimum relevance. Table 3.5 gathers this information on the chosen control points.*

Table 3.5: Control relevance points on NO_2 hourly concentration thresholds for the protection of human health.

Control Points for a Relevance Function ϕ			
y_k: LNO2 concentration values		$\phi(y_k)$	$\phi'(y_k)$
low concentration:	$\ln(3\mu g/m^3)$	0.0	0.0
annual mean guideline:	$\ln(40\mu g/m^3)$	0.5	0.0
limit threshold:	$\ln(150\mu g/m^3)$	1.0	0.0
alert limit threshold:	$ln(400\mu g/m^3)$	1.0	0.0

As we have previously mentioned, if not specified otherwise, the derivatives of the control points are assumed by our algorithm to be zero. This makes each control point, a critical point, i.e. a local maximum or minimum. However, different derivatives can be associated to each control point, specially if we intent to specify a particular increase or decrease. This is the case of the annual mean threshold, whose role is only indicative of an increase. Thus, we decided to give this point a positive derivative. This will be reflected in smoother relevance growth from the lower LNO2 *concentration values to the higher* LNO2 *concentration values.*

Once defined the set of control points (cf. Table 3.5), we now proceed to their interpolation, in order to obtain a relevance function on NO_2 emissions predictions. Figure 3.10 presents two resulting relevance functions ϕ: one by our pchip *interpolation algorithm and other by* splinefun*, a standard cubic interpolation algorithm available on R software (R Development Core Team, 2010).*

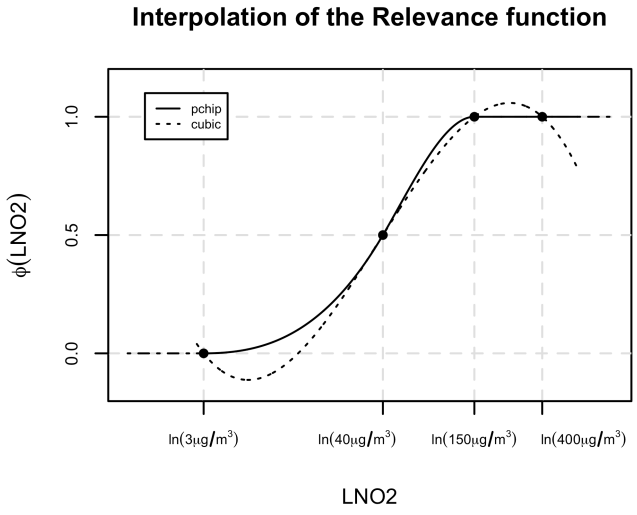


Figure 3.10: A relevance function for the prediction of NO_2 emissions obtained by interpolation: pchip vs. cubic.

As it is possible to observe, the pchip *interpolation produced the relevance function that better suits the application goals. It is our objective to reproduce the user input data with as much*

accuracy as we can. Cubic spline interpolation does not allow us such control over the function. Moreover, for this particular case, the cubic spline interpolation did not confine the function to the stipulated relevance $[0, 1]$ *interval scale. This drawback is addressed in our approach by using appropriate derivates at the control points.*

3.4.2 Identification of Bumps of Relevance

Once we have a relevance function ϕ completely defined, we need to proceed to the bumps identification. Finding the appropriate points to split in the target domain $\mathcal{Y}$, so that we obtain a partition of quasiconcave function, might be a hard and non-trivial task (Hendrix and Toth, 2010). We should stress that there is not a straightforward correspondance between the interpolation points and our bump partition points. Nevertheless, pchip is a monotone piecewise interpolation algorithm and this gives us a head start regarding the bumps identification. In Algorithm 3.2 we show how this process is done based on a relevance function ϕ previously built by pchip.

The bumps algorithm starts by initializing the bumps partition with a first bump $\mathcal{B}_0 = \langle -\infty, -\infty \rangle$. Afterwards it narrows the set of control points to critical points, i.e. those that have a zero derivative. Recall that the definition of each bump requires two values: the left bound b^-, which is a global minimum regarding the non-decreasing range of the bump; and a global maximum b^*. As such, only critical points are worth to analyse. Stepping through each point, the algorithm updates the bumps partition whenever there is a change of relevance value. If there is an increase and was ready to close a bump, then closes it and opens a new one with a new left bound. On the contrary, if there is a decrease and it was on an open bump, then updates the current bump maximum. Whatever is the case, left bounds or maximum values with constant relevance are averaged. The existence of the last bump $\mathcal{B}_{r+1} = \langle +\infty, +\infty \rangle$ is required in theory, but it does not have to be explicitly created.

Example 3.6. Identification of bumps: prediction of NO_2 emissions.

Regarding the relevance function obtained for NO_2 *emissions prediction problem, presented in the previous Example 3.5, three bumps are identified in the bumps partition of the target variable domain. One for each bound of the domain of the target variable and one representing the action*

Algorithm 3.2 bumps(ϕ, S): bumps of relevance defined by the function ϕ.

Input: a relevance function $\phi\colon \mathcal{Y} \to [0,1]$,
set of control points $S = \{\langle y_k, \phi(y_k)\rangle\}_{k=1}^{s}$ with $y_1 < y_2 < \cdots < y_s$.
Output: $\mathcal{P}_\phi(\mathcal{Y})$: a partition of $\mathcal{Y}$ into bumps of relevance.

1: $b_0^- \leftarrow b_0^* \leftarrow -\infty$ // initialize bumps partition
2: $S' \leftarrow \{\langle y_k, \phi(y_k)\rangle \in S : \phi'(y_k) = 0\}$ // select the critical points
3: $r \leftarrow 1$; $l \leftarrow 1$; $s' \leftarrow |S'|$; inBump $\leftarrow$ **true**
4: **for** $i \leftarrow 1$ **to** $s' - 1$ **do**
5: **if** $\phi(y_i) < \phi(y_{i+1})$ **and** !inBump **then**
6: $r \leftarrow r + 1$ // start a new bump
7: $b_r^- \leftarrow \sum_{j=l}^{i} y_j/(i - l + 1)$
8: inBump $\leftarrow$ **true**
9: **else if** $\phi(y_i) > \phi(y_{i+1})$ **and** inBump **then**
10: $b_r^* \leftarrow \sum_{j=l}^{i} y_j/(i - l + 1)$ // close current bump
11: inBump $\leftarrow$ **false**
12: **end if**
13: **if** $\phi(y_i) \neq \phi(y_{i+1})$ **then**
14: $l \leftarrow i + 1$
15: **end if**
16: **end for**
// if there was any change, close the last bump
17: **if** $r > 0$ **then**
18: **if** inBump **then**
19: $b_r^* \leftarrow \sum_{j=l}^{s'} y_j/(s' - l + 1)$
20: **else**
21: $r \leftarrow r + 1$
22: $b_r^- \leftarrow \sum_{j=l}^{s'} y_j/(s' - l + 1)$
23: $b_r^* \leftarrow +\infty$
24: **end if**
25: **end if**
26: **return** $\mathcal{P}_\phi(\mathcal{Y}) = \left\{\langle b_i^-, b_i^*\rangle\right\}_{i=0}^{r}$

bump, where the change relevant/irrelevant values occurs. Figure 3.11 illustrates the identification of these bumps over the relevance function of LNO2.

The bumps partition defined for LNO2 *concentration values is*

$$\mathcal{P}_\phi(\mathcal{Y}) = \{\langle -\infty, -\infty\rangle, \langle 1.1, 5.5\rangle, \langle +\infty, +\infty\rangle\}.$$

The maximum value of action bump $b_1^* = 5.5$ is the result of the average of the two alert values indicated has having maximum relevance. Moreover, according to this bumps partition, the maximum admissible loss established for the action bump is $b_1^\Delta \approx 8.8$. As this is the only action bump existing in the domain, the same value is extrapolated outside the range, i.e. $b_0^\Delta = b_2^\Delta = b_1^\Delta$.

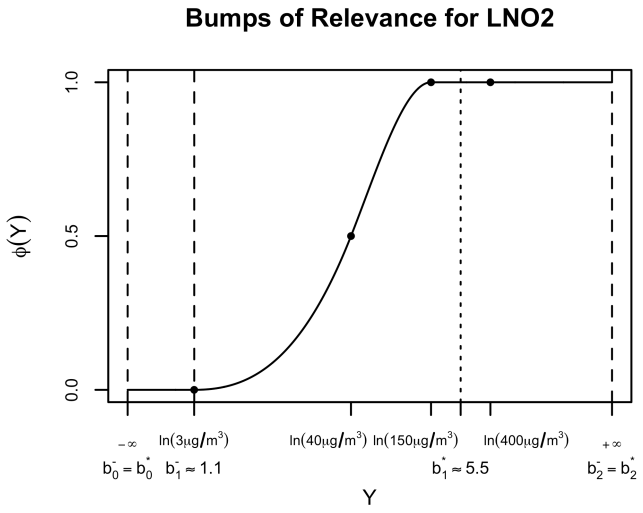


Figure 3.11: The set of bumps identified by bumps algorithm for the prediction of NO_2 emissions. This identification is based on the set of control points given to pchip algorithm.

There are many interpolation techniques, different from *piecewise cubic Hermite interpolation*, that could have been used with the same goal. Nevertheless, the most effective ones are based on piecewise cubic polynomials.

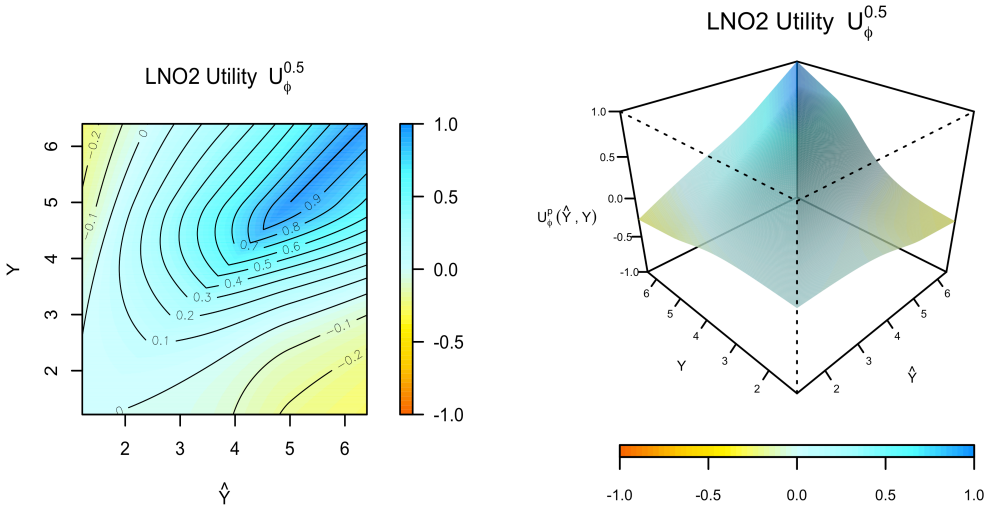
3.5 Illustrative Utility Surfaces

In this section, we present a series of utility surfaces obtained for prediction problems with different relevance functions of the target variable. Our goal is to explain how our proposed utility function represents the domain preference bias in terms of benefits and costs. Through these illustrative surfaces, the two criteria that guide the assessment of benefits and costs should become more perceptible. In particular, we are referring to the accuracy of the predicted value and the correctness of the predicted action. For this illustration we will use the NO_2 emissions prediction problem and a set of artificially generated artificial problems.

3.5.1 Outdoor Air Pollution Prediction Problem

The NO_2 Emissions prediction problem has been partially addressed in Section 3.4. Namely, we have defined a relevance function ϕ in Example 3.5 and derived the bumps partition $\mathcal{P}_\phi(\mathcal{Y})$ in Example 3.6. Hence, we are now able to proceed to the analysis of the resulting utility surface.

In Figure 3.12, we present the first utility surface obtained for the NO_2 Emissions prediction problem, with the penalization costs factor $p = 0.5$.



(a) utility isometrics

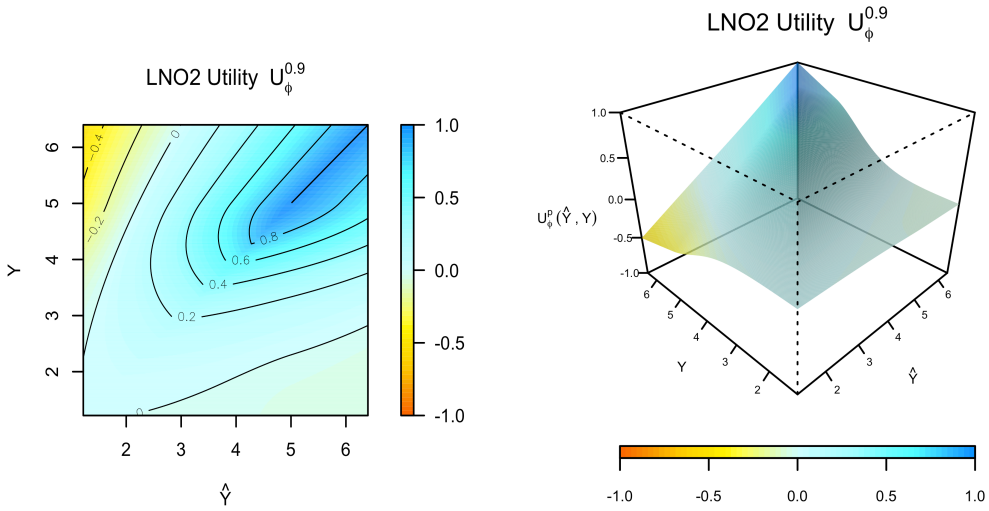
(b) utility surface in benefits perspective

Figure 3.12: An utility surface for the the prediction of NO_2 emissions obtained with the relevance function ϕ of Figure 3.10, with $p = 0.5$.

The surface shown in Figure 3.12b describes how the utility values evolve as a function of the true and predicted values. With a closer inspection of the utility isometrics of the surface, i.e. the lines that share the same value of utility, shown in Figure 3.12a, we obtain a better understanding regarding the costs and benefits associated to this application. Near the diagonal where true and predicted values are equal, we have a positive utility that grows fast as we reach higher values of both the predicted and true values (top right corner). These are the values with higher relevance. At the top left and bottom right corners we get costs, i.e. negative utility values. These areas correspond to inaccurate predictions leading to incorrect actions or predictions where the loss value goes beyond the maximum admissible loss of 8.8 (cf. Example 3.6). As the domain is characterized by having one action (bump) only, that is alarm high `LNO2` concentration values, and the admissible loss is large, costs never get too high for the range of values considered in the graphs. This explains why the utility surface is mainly positive.

We generated a second utility surface for the same setup, but with $p = 0.90$. The resulting surface, shown in Figure 3.13, is very similar to the previous one (cf. Figure 3.12), but with the costs more pronounced. The reasoning is that by setting $p = 0.90$, *opportunity costs* are considered more serious than *false alarms*. This means that missing the prediction of a harmful NO_2 emission is considered to cause more damages than making a false prediction of one. As more weight is given to the true value relevance component in the costs, *false alarms* get even less punished and are assigned smaller costs. This results in an utility surface that is almost constant at the bottom

right area but sharper at the top left area, which corresponds to the *opportunity costs* area. Thus, in comparison with $U_{\phi}^{0.5}$, the surface $U_{\phi}^{0.90}$ exhibits higher costs associated to large errors over true relevant values and much smaller costs associated with *false alarms*.



(a) utility isometrics

(b) utility surface in benefits perspective

Figure 3.13: An utility surface for the the prediction of NO_2 emissions obtained with the relevance function ϕ shown in Figure 3.10, with $p = 0.90$.

Previously, on Section 3.2.2, we have argued about the inadequacy of the standard errors metrics for addressing regression problems with non-uniform benefits/costs. We have been using the NO_2 emissions prediction problem throughout this chapter as an example of such problems. In the Example 3.1 (page 70), we have shown that standard error metrics are inadequate for model selection, in the context of this application. In the following example, we complete the referred example showing that our proposed utility-based performance metrics provide a more effective comparison of regression models than the standard error metrics.

Example 3.7. Performance estimates with standard error metrics and utility-based metrics: prediction of NO_2 emissions.

We have estimated the performance of the two hypothetical models M_1 *and* M_2*, first presented in the Example 3.1 (page 70), with the measures* $MU_{\phi}^{0.5}$ *and* $MU_{\phi}^{0.90}$*. According to the application goals, we now obtain a correct ranking of the models as shown in Table 3.6.*

Contrary to the standard error metrics, we get different performance estimates for the two models using our utility-based metrics. Even though both models obtain an average positive utility value,

which means that their predictions, on average, yield to a benefit, we get an accurate feedback on their relative performance and ranking. Model M_1 *achieved a smaller utility score than model* M_2*, in accordance to the application bias. With* $\mathrm{MU}_{\phi}^{0.90}$, false alarms *get less penalized than* opportunity costs. *In this context, the difference between the performance estimates obtained by the two models increases.*

Table 3.6: Standard error vs utility-based performance estimates of two artificial sets of predictions for NO_2 emissions.

	Estimated Performance			
	MAD	**MSE**	$\mathbf{MU}_{\phi}^{\mathbf{0.5}}$	$\mathbf{MU}_{\phi}^{\mathbf{0.9}}$
M_1	0.402	0.460	0.558	0.558
M_2	0.402	0.460	0.610	0.613

3.5.2 Artificial Prediction Problems

With the goal of illustrating different surfaces, we generated a set of artificial problems described by a random continuous variable combined with pre-specified relevance functions ϕ associated to it. As continuous target variable Y we used a random sample with 1000 observations obtained from a normal probability density distribution $Y \sim \mathcal{N}(0, 5)$ (cf. Figure 3.14).

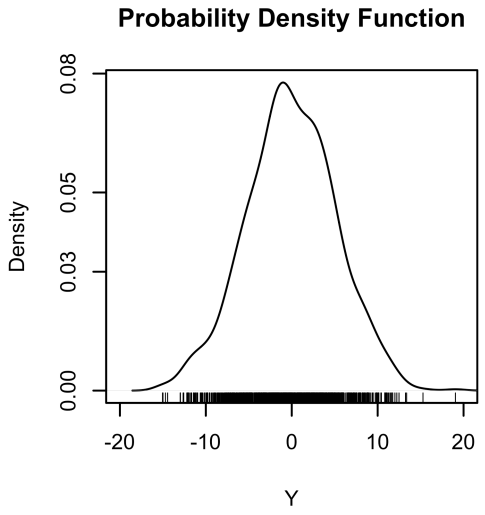


Figure 3.14: The probability density distribution function of a random continuous variable $Y \sim \mathcal{N}(0, 5)$.

With respect to the relevance functions ϕ, we defined them such that the bump regions appear at different ranges of the target Y and with different sizes. Our objective is to create target

prediction problems with a diverse set of relevance functions. As penalization costs factor we have used $p = 0.75$ to increase the difference between the different types of costs.

Using these settings, we defined 4 problems: `AllValues`, `CommonValues`, `RareValues` and `SpecificValues`. In the next sections we describe each of these problems and present their respective utility surfaces.

3.5.2.1 `AllValues` Problem

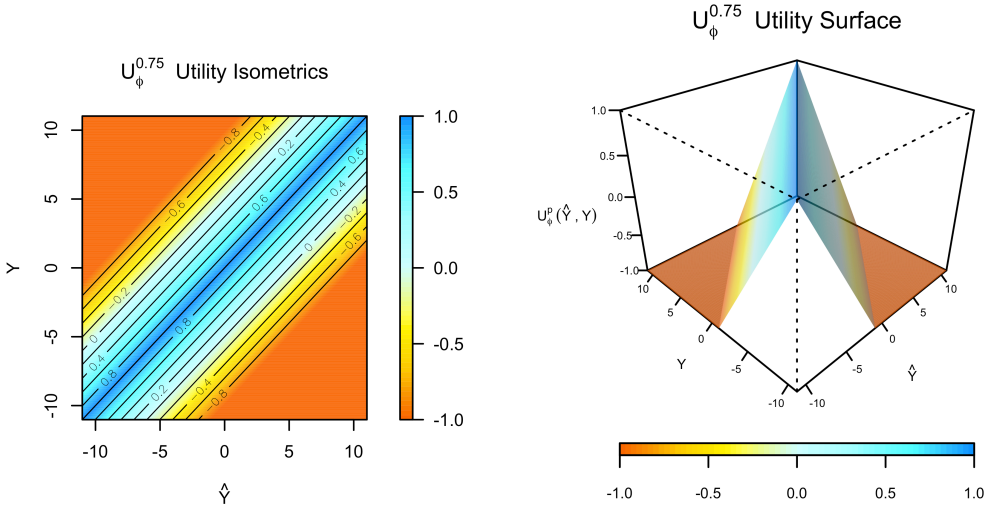
In the `AllValues` problem, all the values in the domain of the continuous target variable Y are assigned the maximum relevance, i.e. $\phi(Y) = 1$. As we have mentioned earlier, this is the case of standard regression where the relevance of the values of the target variable is uniform across its domain. The relevance function ϕ is constant and $\mathcal{Y}$ has a single bump. The bump partition defined for this problem is $\mathcal{P}_\phi(\mathcal{Y}) = \{\langle -\infty, -\infty \rangle, \langle +\infty, +\infty \rangle\}$.

With a constant relevance function, in this case $\phi(Y) = 1$, our utility function is reduced (cf. Section 3.3.3) to

$$U_1(\hat{y}, y) = 1 - 2\,\Gamma_\varepsilon(\hat{y}, y) \tag{3.24}$$

where Γ_ε is a bounded loss function based on the maximum loss tolerance ε. In cases like this, we suggest ε to be estimated as the standard error of Y. However, for illustration purposes, we will use $\varepsilon = 10$. The function U_1 is a linear transformation of the prediction error provided by some loss function L into a $[-1, 1]$ range of utility values. This transformation is such that loss values of zero are mapped to the maximum utility value and, all the loss values above or equal to the selected value of ε are mapped to minimum utility value. As the loss increases towards ε, so it decreases the utility. Because the relevance of the values is constant, they have no influence on this process, neither it has the p parameter. The resulting utility surface is shown in Figures 3.15a and 3.15b.

This utility surface is similar in behaviour to what we get in a standard regression setting. The shape is a consequence of the loss function used in the bounded loss function Γ_ε, which was the absolute deviation. Moreover, in the isometrics presented in Figure 3.15a, we can see the impact of the maximum loss tolerance $\varepsilon = 10$. This threshold defines the size of the band where "potentially useful" predictions fit. Positive utility values are obtained near the main diagonal, i.e. for predictions leading to an absolute loss below ε. The utility of each prediction is then evaluated by Equation 3.24, which scales each loss with respect to the maximum loss ε. All the other predictions, where the loss value is above ε, are considered to be errors, and thus are associated with the maximum cost value, that is -1. For this particular problem, the error loss is the only criterion having an impact on the resulting utility. If we want to use our proposed framework on a standard regression problem, this is the kind of utility surface we will get.



(a) utility isometrics

(b) utility surface in benefits perspective

Figure 3.15: An utility surface for `AllValues`.

3.5.2.2 `CommonValues` Problem

The relevance function for the `CommonValues` problem is presented in Figure 3.16. This function defines the relevant values as the most common values of the target variable according to the pdf shown in Figure 3.14. From the relevance function ϕ, the following bump partition is obtained: $\mathcal{P}_\phi(\mathcal{Y}) = \{\langle -\infty, -\infty\rangle, \langle -20, 0\rangle, \langle 20, +\infty\rangle\}$.

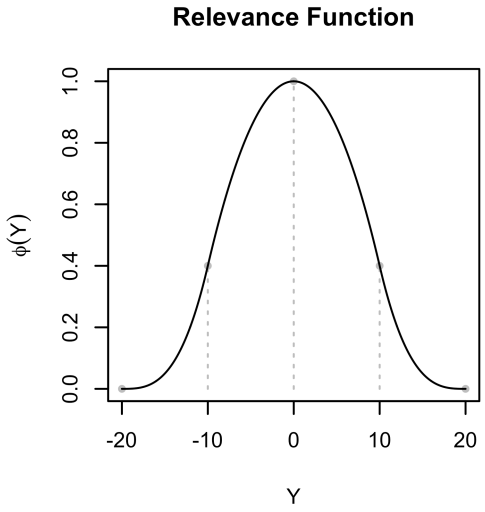
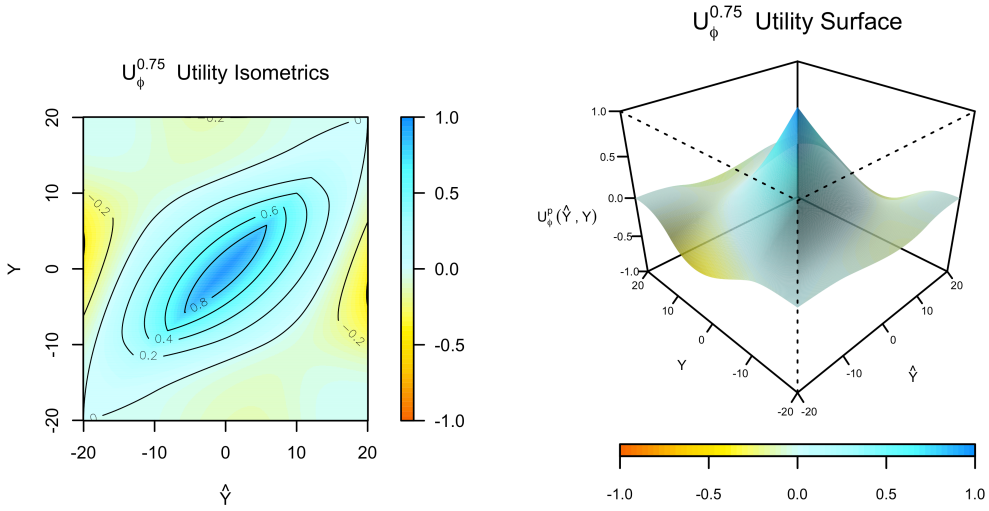


Figure 3.16: The relevance function of artificial problem `CommonValues`.

Unlike in `AllValues` problem, the relevance function is not uniform. In this context, the utility scores take into account two aspects: the loss value of the predicted values and the correctness

of the action associated with these values. Related to the former of these aspects we have the maximum loss tolerance of the sole "action bump" $\mathcal{B}_1$ that is equal to 40. With respect to the second aspect, according to the selected relevance function, only the predictions involving true or predicted values within bump $\mathcal{B}_1$ achieve more significant values of utility, as shown in Figure 3.17a.



(a) utility isometrics

(b) utility surface in a benefits perspective

Figure 3.17: An utility surface for `CommonValues`.

Looking at the utility isometrics shown in Figure 3.17a, we can observe the clear relationship between the highest relevance region of the true values and the highest values of both benefits and costs. From the perspective of true values, we have two patterns of utility: one located at the central values where the most relevant values are; and the other located at the extremes where the less relevant values appear. For the highly relevant values at the center of the Y variable distribution we observe a much wider variability in utility scores. The reason is that high relevance of true values may yield to both higher benefits but also higher costs. Still, because this problem contains a single event bump, predictions can not incur on *confusing events* and thus the highest costs are only attainable with very inaccurate predictions due to the large value of the maximum admissible loss.

3.5.2.3 `RareValues` Problem

The relevance function ϕ defined for the `RareValues` problem and presented in Figure 3.18, establishes that, contrary to the previous problem, rare values of the target variable Y are the most relevant.

From the relevance function ϕ, we obtain the following bumps partition: $\mathcal{P}_\phi(\mathcal{Y}) = \{\langle -\infty, -20\rangle, \langle 0, 20\rangle\}$.

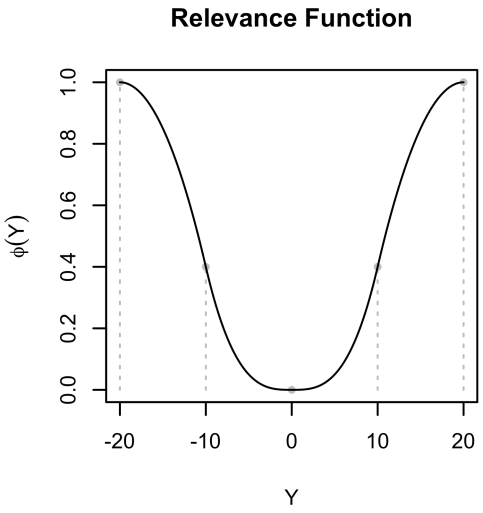
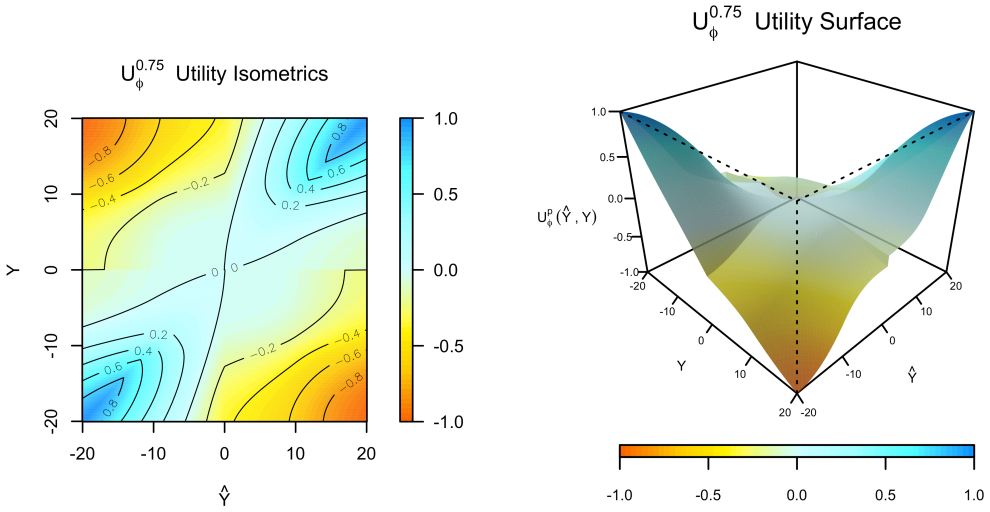


Figure 3.18: The relevance function for artificial problem `RareValues`.

Once again, the utility isometrics, shown on Figure 3.19a, confirm the association of higher benefits and costs to regions of higher relevance of the true and/or predicted values. The maximum admissible loss is equal for both bumps, and it is $b_0^{\Delta} = b_1^{\Delta} = 40$.



(a) utility isometrics

(b) utility surface in costs perspective

Figure 3.19: An utility surface for `RareValues`.

The utility surface obtained for this problem exhibits some similarities with the previous problem `CommonValues`. Their similarity arises from the complementarity of the relevance definition for the two problems. Here again, and from the true values perspective, the utility surface assumes two main patterns: one at the extremes, where the target variable assumes the most relevant values, and other at center, where the target variable is less relevant. Values near the center of

distribution of Y have zero relevance and thus any prediction for these values will not achieve a significant benefit. Nevertheless, as the predicted values become more extreme, so it increases their associated cost values. In effect, for this problem, depending on the relevance of the true and the predicted value, it is possible to incur in *false alarms*, *opportunity costs* or *confusing events*. Because $p = 0.75$, the predictions of large positive or large negative values for true values that are irrelevant (false alarms), is less penalized than opportunity costs. The worst cost scenario is achieved when a large positive value is confused with a large negative value or vice-versa. As a consequence, the isometrics for extreme values (positive or negative) show the largest variation in terms of benefits and costs.

3.5.2.4 `SpecificValues` Problem

The `SpecificValues` problem defines that relevant values belong to specific ranges of the target variable Y. The used relevance function ϕ is shown in Figure 3.20.

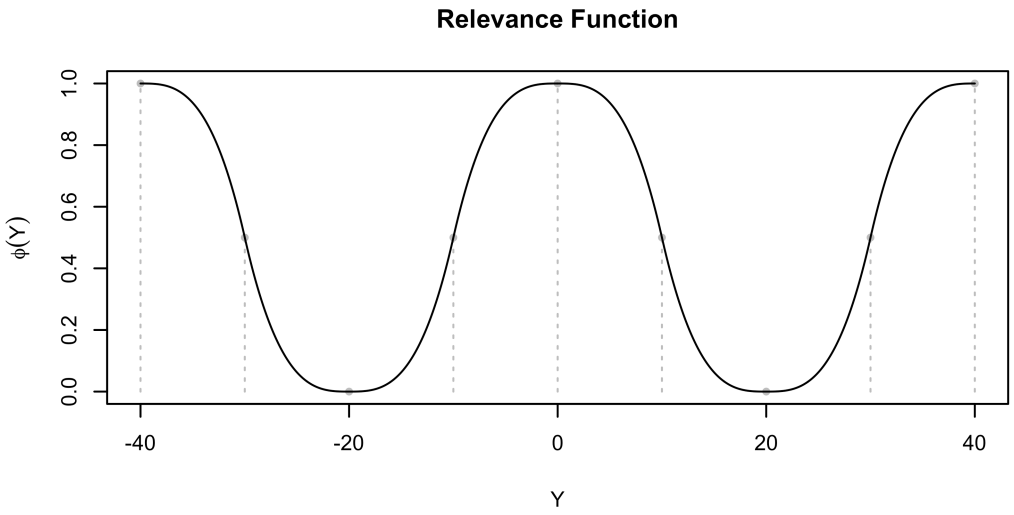
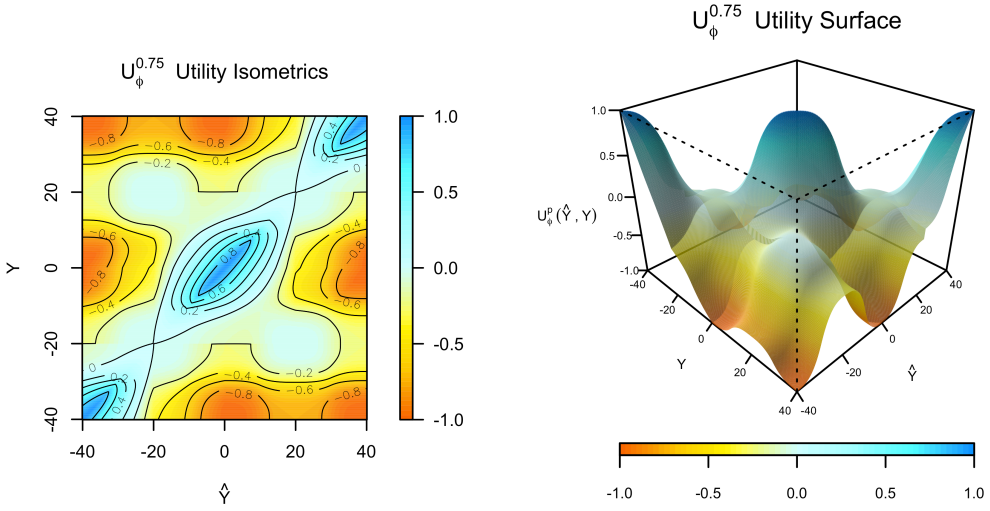


Figure 3.20: The relevance function for artificial problem `SpecificValues`.

From the relevance function ϕ, we obtain the following bumps partition: $\mathcal{P}_\phi(\mathcal{Y}) = \{\langle -\infty, -40\rangle, \langle -20, 0\rangle, \langle 20, 40\rangle\}$. The maximum admissible loss obtained for all the three bumps is equal, i.e. $b_0^\Delta = b_1^\Delta = b_2^\Delta = 40$.

From the observation of the utility isometrics presented in Figure 3.21a, it is possible to identify in the Y direction, i.e. horizontally, two different types of patterns of utility. The first type of pattern appears at both extremes and at the center of Y and it is related with the three bumps: $\mathcal{B}_0$, $\mathcal{B}_1$ and $\mathcal{B}_2$. At these three regions, we observe high benefits or high costs that result from the high relevance of the true value and the proximity of other bumps with high relevance. This



(a) utility isometrics

(b) utility surface in costs perspective

Figure 3.21: An utility surface for `SpecificValues`.

latter fact increases the chances of *confusing events*, the worst type of error for these scenarios. The second pattern refers to the regions between bumps. These are regions characterized by low relevance of values for which any accurate prediction leads to insignificant benefits. Still, even within these areas if a highly relevant value is predicted we can incur in significant costs, as these situations correspond to *false alarms*. The overall impact of these costs on the surface depends on the penalization parameter p. In this problem, we have used $p = 0.75$ and thus *opportunity costs* are considered more serious than *false alarms* as it can be confirmed by the utility isometrics.

Comparing the figures of this problem with the corresponding graphs for the problem `AllValues`, where the relevance was uniform across the domain of the target variable, we can see how different are the preference biases of this type of applications when compared to the more standard regression tasks illustrated by problem `AllValues`. The proposed utility-based evaluation framework is much more consistent with the application goals for non-uniform regression problems.

3.6 Experimental Analysis

We have presented an utility-based evaluation methodology that is able to handle applications with non-uniform costs across the domain of a continuous target variable. In this section, we test the sensitiveness of such methodology in the task of identifying the best models for this kind of applications, when compared with another evaluation methodology based on a standard error metric.

We claim that without our proposed utility-based metrics, we can only obtain suboptimal results in terms of comparing models in the context of our target applications. Through a set of experiments we will show that standard evaluation metrics are not always able to choose the best models. We demonstrate this by building a sort of improved versions of the models used in the experiments. These are obtained by modifying the set of originally made predictions such that their performance is made more adequate to the applications preference biases. As so, one expects that they are ranked first. However, as we will show that is not the case if we use standard error metrics.

In this context, we claim that the use of standard error metrics on applications with non-uniform costs may lead to sub-optimal, or even wrong decisions when selecting/comparing alternative regression models, which can be critical for some domains. Moreover, we will show that our proposal can overcome these limitations.

3.6.1 Prediction Problems

In order to accomplish our objective, two kinds of experiments were performed. The first experiment was conducted on real data, namely on the NO_2 concentration data set described in Section 3.2.1 (page 67). For the experiments we used all the domain knowledge that we had. The used relevance function ϕ was the same shown in Figure 3.10 (page 98). Concerning the utility surface, we established $p = 0.75$ to make *opportunity costs* more serious than *false alarms* like in the utility surface of Figure 3.13 (page 103).

With respect to the second experiment, our goal was to test the sensitivity of the results to the choice of relevance function. We have generated an artificial data set with a continuous target variable, and we have specified several diversified relevance functions for this same data. This was the same process used in Section 3.5. In particular, we are referring to the problems: `AllValues`, `RareValues`, `CommonValues` and `SpecificValues`. The generated artificial data set has 1000 examples and is described by four attributes - one nominal and three continuous, and a target continuous variable.

Our goal in this second set of experiments is to confirm that our utility measure will be able to choose the best model for a diverse set of relevance function (i.e. non-uniform regression settings). In terms of the balance between the different types of errors we have set $p = 0.5$, thus making *false alarms* equally important as *opportunity costs*.

3.6.1.1 Methodology

With the goal of avoiding any bias on our conclusions concerning the used modelling techniques, we have selected four quite different approaches. For all of these techniques we have used the implementations available on the R software environment (R Development Core Team, 2010). The selected methods were:

- regression trees (cart) - we used the implementation available through package `DMwR` (Torgo, 2010), with default setting `se=1`.
- support vector machines (svm) - we used the implementation available in the package `e1071` (Dimitriadou et al., 2010), width default setting `cost=1` and radial-basis kernel.
- multivariate adaptive regression splines (mars) - we used the implementation available in the package `earth` (Milborrow et al., 2010), with default setting `degree=1` and `thresh=0.001`.
- random forests (randomF) - we used the implementation available in `randomForest` package (Liaw and Wiener, 2002), with default setting `ntree=500` and `nodesize=5`.

No extensive parameter tuning was carried out, as top performance is not the goal here - our goal is to compare the model rankings produced by different evaluation metrics under different setups.

None of the alternative models we are considering in our experiments optimizes any of our utility-based metrics. In this context, we incur the danger of concluding that all models perform equally bad in terms of the applications goals, which would not allow us to conclude anything concerning the eventual advantages of our metrics. In order to avoid this problem we wanted to ensure that among the candidate models there were alternatives that were clearly better in terms of being able to optimize utility. These models should come up as the best models if the used metrics have any value. To obtain these "optimal models" we generated artificial sets of predictions based on the predictions of the "standard" models. Namely, we created these artificial predictions by tweaking the prediction errors of our original four modelling techniques in such a way that the errors are artificially re-allocated to the test cases that improve the overall utility score. By proceeding this way, we reach to a set of artificial set of predictions that have exactly the same set of error amplitudes as the base models, but with a higher utility value because the smallest errors were re-allocated to more relevant true values. [⋆7]. In our experiments we named these artificial sets of predictions improv. $<base\_technique>$ (e.g. improv.cart). From the perspective of the goals of our target applications, these improved models should appear ahead of the base models (or at most at the same position in the unlikely event that the base model already achieves this "improved"

[⋆7] This strategy is similar to the one followed on the illustrative Example 3.1 given on Section 3.2 (page 65).

performance). A failure to rank these improved models on top would mean the failure of our proposal.

All the alternative models were evaluated using three different evaluation statistics: MU, NMU and NMAD. The values of these statistics have different meanings. Concerning the MU evaluation metric, positive values indicate that the model is useful on average, while negative values indicate that the model usually issues predictions that represent costs (i.e. negative benefits). The NMU reflects the same results but in the interval $[0, 1]$ as it is the normalized version of MU. For comparison purposes, we have also evaluated the performance of the models by the NMAD evaluation metric, selected as a "representative" of a standard evaluation metric that consists of a normalized version of MAD to the interval $[0, 1]$, as shown in Equation 3.25. All the evaluation metrics use, as loss function, the absolute deviation.

$$\text{NMAD} = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{\sum_{i=1}^{n} |\widetilde{Y} - y_i|} \tag{3.25}$$

where $\widetilde{Y}$ is the median of the target variable Y.

The performance of each modelling technique for all the statistics mentioned above was estimated using a stratified 10x10-fold cross validation process. The statistical significance of the difference between the score of each modelling technique and the one ranked as the best, was measured using the *Student paired t-test*. For each statistic we provide the mean (μ) and the standard deviation (σ) obtained over the 100 repetitions of the 10x10-fold cross validation process.

3.6.2 Experimental Results

The model rankings obtained for the Outdoor Air Pollution application are presented in Table 3.7. For this particular data set, considering its small number of cases, we chose to run a stratified 1x10-fold cross validation process to obtain the performance estimates.

From a first analysis of the results, it is possible to notice that our utility-based metrics (MU and NMU) and NMAD do not agree on the top ranked modelling technique. Moreover, as expected [*8], the NMAD statistic is not able to distinguish between the improved versions of the models and the original ones. According to NMAD randomF is the best model for this problem. However, randomF is not a well positioned modelling technique according to both our utility metrics, with a performance that is significantly worse than the top model according to these two metrics. Still, according to MU, the performance of randomF is positive, which means that on average its

[*8] This is the same case of models M_1 and M_2 shown in Example 3.1 (page 70).

Table 3.7: Performance ranking obtained by utility-based metrics and a standard error metric for different algorithms in the prediction of NO_2 emissions.

Performance Estimates Ranking			
Data Set	MU : $\mu \pm \sigma$	NMU : $\mu \pm \sigma$	NMAD : $\mu \pm \sigma$
NO_2 Emission	improv.randomF 0.51417 ± 0.03604	improv.randomF 0.75709 ± 0.01802	randomF 0.61803 ± 0.07345
	improv.svm 0.51264 ± 0.03573	improv.svm 0.75632 ± 0.01787	improv.randomF 0.61803 ± 0.07345
	improv.mars 0.51222 ± 0.03617	improv.mars 0.75611 ± 0.01809	mars • 0.65046 ± 0.10225
	improv.cart •• 0.50609 ± 0.0358	improv.cart •• 0.75305 ± 0.0179	improv.mars • 0.65046 ± 0.10225
	svm •• 0.48409 ± 0.03248	svm •• 0.74205 ± 0.01624	svm • 0.66435 ± 0.05996
	randomF •• 0.48344 ± 0.03268	randomF •• 0.74172 ± 0.01634	improv.svm • 0.66435 ± 0.05996
	mars •• 0.48009 ± 0.03302	mars •• 0.74005 ± 0.01651	cart •• 0.75887 ± 0.05341
	cart •• 0.46883 ± 0.03303	cart •• 0.73441 ± 0.01651	improv.cart •• 0.75887 ± 0.05341

Through 1x10 cross-validation, the *paired t-test* asserts that the difference to the best ranked modelling technique is:
• significant at 95% confidence level;
•• significant at 99.9% confidence level.

predictions lead to a benefit. In effect, in this application all modelling techniques achieved a globally good utility score. This is explained by the overall positive utility surface of this problem as presented in Section 3.5.1 (see page 101). Independently of these observations, our experiments show that our metrics are able to identify the best models (which were artificially made the best) from the perspective of the user preference biases for this application. Moreover, according to our metrics, the best model is improv.randomF. Still, more important than this observation is the fact that our metrics were able to rank at the top all improved variants of the base models, which provides strong evidence concerning their ability to identify models that perform better in terms of the goals of this application.

We now move into the second group of experiments where we try to test the sensitivity of our metrics to the type of relevance function. The results on the artificially generated problems are shown in Tables 3.8, 3.9, 3.10 and 3.11.

The first thing we should note on these tables of results is that the NMAD and the utility-based metrics never agree on the overall ranking of the models for all of the problems. Even more importantly, we note that once more, and for all different types of relevance functions[⋆9], our metrics are able to rank the improved versions above the base models. This means that under all these experimental settings the behaviour of our metrics was in accordance to our hypothesis.

The results obtained for problem `AllValues` are shown in Table 3.8. This problem is a simulation of a standard regression scenario with its uniform relevance. In this context, it is not surprising

[⋆9] With the exception of problem `AllValues` for obvious reasons that shall be explained below.

Table 3.8: Performance ranking obtained by utility-based metrics and a standard error metric for different algorithms in the `AllValues` problem.

Performance Estimates Ranking			
Data Set	MU : $\mu \pm \sigma$	NMU : $\mu \pm \sigma$	NMAD : $\mu \pm \sigma$
`AllValues`	mars -0.87263 ± 0.0363	mars 0.06369 ± 0.01815	randomF 0.99584 ± 0.00745
	improv.mars -0.87263 ± 0.0363	improv.mars 0.06369 ± 0.01815	improv.randomF 0.99584 ± 0.00745
	cart -0.87427 ± 0.03777	cart 0.06286 ± 0.01889	svm • 0.99722 ± 0.00533
	improv.cart -0.87427 ± 0.03777	improv.cart 0.06286 ± 0.01889	improv.svm • 0.99722 ± 0.00533
	randomF • -0.88117 ± 0.02974	randomF • 0.05941 ± 0.01487	mars • 0.99768 ± 0.00392
	improv.randomF • -0.88117 ± 0.02974	improv.randomF • 0.05941 ± 0.01487	improv.mars • 0.99768 ± 0.00392
	svm •• -0.88798 ± 0.03199	svm •• 0.05601 ± 0.016	cart •• 0.99994 ± 0.00015
	improv.svm •• -0.88798 ± 0.03199	improv.svm •• 0.05601 ± 0.016	improv.cart •• 0.99994 ± 0.00015

Through 10x10 cross-validation, the *paired t-test* asserts that the difference to the best ranked modelling technique is:
• significant at 95% confidence level;
•• significant at 99.9% confidence level.

Table 3.9: Performance ranking obtained by utility-based metrics and a standard error metric for different algorithms in the `CommonValues` problem.

Performance Estimates Ranking			
Data Set	MU : $\mu \pm \sigma$	NMU : $\mu \pm \sigma$	NMAD : $\mu \pm \sigma$
`CommonValues`	improv.cart 0.65077 ± 0.02346	improv.cart 0.82539 ± 0.01173	randomF 0.99584 ± 0.00745
	cart •• 0.65047 ± 0.02351	cart •• 0.82523 ± 0.01175	improv.randomF 0.99584 ± 0.00745
	improv.mars 0.65029 ± 0.02355	improv.mars 0.82514 ± 0.01178	svm • 0.99722 ± 0.00533
	improv.randomF • 0.64977 ± 0.02363	improv.randomF • 0.82489 ± 0.01182	improv.svm • 0.99722 ± 0.00533
	mars •• 0.64969 ± 0.02361	mars •• 0.82484 ± 0.01181	mars • 0.99768 ± 0.00392
	randomF •• 0.64886 ± 0.02367	randomF •• 0.82443 ± 0.01184	improv.mars • 0.99768 ± 0.00392
	improv.svm •• 0.64831 ± 0.02355	improv.svm •• 0.82416 ± 0.01177	cart •• 0.99994 ± 0.00015
	svm •• 0.64725 ± 0.02361	svm •• 0.82362 ± 0.0118	improv.cart •• 0.99994 ± 0.00015

Through 10x10 cross-validation, the *paired t-test* asserts that the difference to the best ranked modelling technique is:
• significant at 95% confidence level;
•• significant at 99.9% confidence level.

that the improved versions of the models are not distinguishable even for the utility metrics. The relevance along the domain of the target variable is constant, and thus the errors have the same value independently of the location within the target variable domain. Still, different rankings are obtained by the utility metrics and the standard regression metric. This happens due to the normalization of the error. Instead of the distance to the median value of Y used by NMAD, the utility metrics use the maximum admissible loss.

Table 3.10: Performance ranking obtained by utility-based metrics and a standard error metric for different algorithms in the `RareValues` problem.

Performance Estimates Ranking			
Data Set	MU : $\mu \pm \sigma$	NMU : $\mu \pm \sigma$	NMAD : $\mu \pm \sigma$
`RareValues`	improv.randomF 0.06402 ± 0.0095	improv.randomF 0.53201 ± 0.00475	randomF 0.99584 ± 0.00745
	improv.svm 0.06388 ± 0.00941	improv.svm 0.53194 ± 0.00471	improv.randomF 0.99584 ± 0.00745
	improv.mars 0.06388 ± 0.00946	improv.mars 0.53194 ± 0.00473	svm • 0.99722 ± 0.00533
	improv.cart • 0.06371 ± 0.00942	improv.cart • 0.53185 ± 0.00471	improv.svm • 0.99722 ± 0.00533
	svm •• -0.01544 ± 0.00327	svm •• 0.49228 ± 0.00163	mars • 0.99768 ± 0.00392
	randomF •• -0.01595 ± 0.00332	randomF •• 0.49203 ± 0.00166	improv.mars • 0.99768 ± 0.00392
	mars •• -0.01685 ± 0.00353	mars •• 0.49158 ± 0.00177	cart •• 0.99994 ± 0.00015
	cart •• -0.01851 ± 0.00356	cart •• 0.49074 ± 0.00178	improv.cart •• 0.99994 ± 0.00015

Through 10x10 cross-validation, the *paired t-test* asserts that the difference to the best ranked modelling technique is:
• significant at 95% confidence level;
•• significant at 99.9% confidence level.

Table 3.11: Performance ranking obtained by utility-based metrics and a standard error metric for different algorithms in the `SpecificValues` problem.

Performance Estimates Ranking			
Data Set	MU : $\mu \pm \sigma$	NMU : $\mu \pm \sigma$	NMAD : $\mu \pm \sigma$
`SpecificValues`	improv.cart 0.68684 ± 0.02175	improv.cart 0.84342 ± 0.01088	randomF 0.99584 ± 0.00745
	cart •• 0.68684 ± 0.02175	cart •• 0.84342 ± 0.01088	improv.randomF 0.99584 ± 0.00745
	improv.mars • 0.68626 ± 0.02191	improv.mars • 0.84313 ± 0.01095	svm • 0.99722 ± 0.00533
	mars • 0.686 ± 0.02191	mars • 0.843 ± 0.01095	improv.svm • 0.99722 ± 0.00533
	improv.randomF • 0.68573 ± 0.02197	improv.randomF • 0.84287 ± 0.01099	mars • 0.99768 ± 0.00392
	randomF •• 0.68521 ± 0.02197	randomF •• 0.84261 ± 0.01098	improv.mars • 0.99768 ± 0.00392
	improv.svm •• 0.68425 ± 0.0219	improv.svm •• 0.84212 ± 0.01095	cart •• 0.99994 ± 0.00015
	svm •• 0.6836 ± 0.02193	svm •• 0.8418 ± 0.01096	improv.cart •• 0.99994 ± 0.00015

Through 10x10 cross-validation, the *paired t-test* asserts that the difference to the best ranked modelling technique is:
• significant at 95% confidence level;
•• significant at 99.9% confidence level.

An interesting aspect of the results achieved for the problems `CommonValues` and `SpecificValues` is that the top ranked model according to MU is also the last model according to NMAD, that is improv.cart. These problems define that the values at the center of the distribution of Y are relevant. The standard regression models are biased to minimize errors metrics such as MSE or MAD (cf. Equation 3.4 and Equation 3.5 - page 69). These predictive error estimators will tend to favour models that have good performance on the most frequent cases, because they are estimators

of the true mean error. On the particular case of problems `CommonValues` and `SpecificValues` more than minimizing the loss, it is important that the prediction stays confined to the interval defined by a bump. In these conditions, `cart` is probably the model whose predictions are more constant around the mean value.

In `RareValues` the high relevance area is on the tails of the distribution of Y. Those values are extreme and rare. As the regression models are optimizing standard error metrics, the performance of the models turns out to be worst for this regression task. The difficulty in predicting accurately the rare extreme values, causes the highest error estimates for the models. The utility estimates obtained for each model decrease in comparison to the previous problem `CommonValues` (cf. Table 3.9). Moreover, all the original (not improved) versions of the model obtain negative average utility estimates. This means that a prediction made by one of these models yields, on average, to a cost.

`SpecificValues` problem defines an area with oscillating relevance that goes across the distribution of the target variable Y. According to its relevance function ϕ, a value can be relevant if it is either an extreme value or a central one. This means that there are predictions that will be penalized, but not as much as in problem `CommonValues` and not as low as in problem `RareValues`. This yields to intermediate utility performance estimates of the models for this problem. Moreover, these intervals of values with changing relevance, had a great impact in the overall ranking of the models.

Overall, these experiments illustrate very clearly the danger of using a standard error metric, like NMAD, for comparing models in this type of applications with non-uniform costs. The conclusions that one may draw from using these standard error metrics may be completely misleading in terms of the application goals. Moreover, these experiments confirmed our hypothesis that our proposed metrics are able to provide a model ranking that is in accordance with the goals of this class of applications with non-uniform costs across the domain of the target variable.

3.7 Conclusions

In this chapter we have presented a new utility-based evaluation methodology. This framework allows a reliable evaluation of regression models in application with non-uniform costs and benefits across the domain of the target variable. The proposed methodology can be used to compare and/or select models in this type of domains.

Our proposal consists of an utility surface defined with the help of the end-user that provides the domain knowledge concerning the application preference biases. This is also the case of

any approach to utility-based learning. The domain-knowledge is incorporated by means of the definition of a relevance function ϕ. The function ϕ is a continuous function that maps the domain of the target variable into a $[0, 1]$ scale of relevance. We define the utility of a prediction as the net balance between the benefits and costs associated with the predicted value and the respective true value. To calculate this utility we take into account the relevance of the true and predicted values and consider two different aspects: (i) the accuracy of the action associated with the predicted value; and (ii) the numeric accuracy of the predicted value.

Our experiments have confirmed the risks of using standard evaluation metrics when comparing models in applications with non-uniform costs. Namely, there are problems where standard metrics are unable to identify the best models, which can be critical for some real-world applications. Moreover, we have shown that these risks can be overcome by the use of our metric, which is able to provide a model ranking that is in accordance to the preference biases of the applications.

As we will see in the next chapters, based on this methodology of utility-based evaluation, we will propose other metrics commonly used in decision-making processes and ranking analysis, thus allowing better model comparisons under different setups. Moreover, by plugging one of these utility-based metrics into our regression system ubaRules, we will be able to obtain models that are optimized according to the preference biases of these applications.

- 4 -

Prediction of Rare Values in Regression

"Familiarity breeds contempt, while rarity wins admiration."
Apuleius (A.D. 123/125 - 170)

Many event-based applications like prediction of ecological or meteorological catastrophes, fraud detection, network intrusions and financial forecasting are particular cases of regression problems with non-uniform costs on the target continuous variable that associate higher costs or benefits with rarity. Rare values are often regarded as outliers and removed from posterior analysis. However, for this type of applications, they are related to the anticipation of a critical phenomenon, and thus should be accurately predicted.

We propose a decision making process for regression that is sensitive to the utility (benefits / costs) of the predictions. Our goal is to be able to use evaluation metrics like precision and recall, which focus the evaluation of predictive models on the cases that really matter, by adapting them for these applications. This adaptation involves taking into account the numeric error of the predictions using the notion of utility we have proposed in the previous chapter.

In non-uniform domains such as the ones we are tackling, using a fixed decision threshold might not be a reliable option. As alternative, we also propose the incorporation of utility-based regression into ranking analysis. The objective is to obtain an accurate ranking of the models towards the prediction of the target event, i.e. the rare values, as a more trustful and interpretable information to the end-user. In particular, we present the visualization of models through Precision-Recall (PR) curves that incorporate instance varying utility. Based on utility, we also derive common ranking metrics, such as $AUC - PR$, MAP11, BFM.

Along the chapter, a real-world problem concerning the prediction of large forest fires is used to better illustrate our proposals. At the end, we conduct an experimental study on two other real-word problems: forecasting of stock market returns and prediction of harmful algae blooms. Experiments on both problems show the advantages of using our proposed utility-based metrics when comparing predictive models in the context of this type of applications.

4.1 Introduction

Several important predictive data mining applications involve handling non-uniform costs and benefits of predictions. This is almost always the case in event-based applications in many domains (e.g. ecology, meteorology, finance, fraud). Many of these tasks are particular cases of regression problems where the continuous target variable values have differentiated importance. These prediction tasks are often related to the anticipation of a critical phenomenon that is inherently continuous and for which an alarm may be triggered by a specific range of values of a continuous target variable. This type of applications require techniques that are able to cope with predictions that have differentiated costs and benefits. In Chapter 3 we have presented and utility-based evaluation framework that can be used to help in addressing these applications.

In this chapter our goal is to address a particular and highly relevant sub-class of non-uniform costs/benefits prediction tasks, which associates higher cost or benefit with rarity. For these applications the most (and often solely) important cases are the ones associated with unusual values of the target variable. That is the case of the task of predicting unusual returns in stock markets (e.g. Torgo and Ribeiro, 2009). For this application a model that is excellent at predicting the most common values of the target variable (the returns of a financial asset), is basically useless. In effect, the most common returns are around zero (or small variations), and these values are not profitable due to the unavoidable transaction costs. This is not a particularity of financial markets. In ecological modelling, for instance, the prediction of harmful algae blooms (e.g. Ribeiro and Torgo, 2008a) is a very relevant task. These algae blooms are associated with rare and unusually high values of abundance for certain algae species. Once again, it is the predictive performance on these rare extreme values that really matters for the end user, as it allows for proper preventive actions to be taken. The common characteristic of these applications is that their main goal is accuracy at predicting values that most statistical analysis would tag as outliers. We are thus facing a task of predicting outlier values of a continuous target variable. Evaluating the ranking performance of regression models on such type of domains is also considered to be a very useful tool (Rosset et al., 2007). Complementarily to an error-based evaluation, a ranking-based evaluation for the comparison of models might be essential to identify the models, which act better as decision makers. These are the models that better discern high relevance values from low relevance values and, thus, that would bring more benefits than costs. In finance domains, for instance, where there is a large number and variety of models it is important to be able to compare their performance in an objective and meaningful way. Hence, as in this case, ranking-based metrics can be more valuable for the end-user.

Most of the cost-sensitive techniques that have been proposed in the literature (e.g. Domingos, 1999; Elkan, 2001; Zadrozny, 2005) address predictive classification tasks. So far, we have shown

that utility-based concepts can also be applied to regression tasks with non-uniform costs. These concepts enabled the proposal of new utility-based metrics, which allow a more effective evaluation of models in the context of the target applications. For rare and extreme values prediction, extreme values have higher relevance comparatively with the rest of the target variable domain. As such, all the predictions made on extreme values are of major importance for the model performance.

In this chapter, we propose new evaluation metrics, specially created for the prediction of rare extreme values of a continuous variable. In order to do so, some of notions given in Chapter 2, in the context of rare values prediction, will be revisited.

Section 4.2 starts by the formulating our target prediction problems and introducing the notation conventions adopted. The proposal for decision-making in regression is made on Section 4.3. In this section, we explain how the notions of precision and recall can incorporate utility. Ranking analysis in regression is presented in Section 4.4. The comparison of models through Precision-Recall (PR) curves sensitive to utility and some utility-based ranking measures are explained in this section. Section 4.5 presents an experimental study conducted over two concrete applications where the target objective is the prediction of rare extreme values: the forecast of future returns of a set of stocks and the prediction of harmful algae blooms. The collected experimental evidence indicates the advantages and usefulness of the proposed evaluation framework. Section 4.6 finishes with conclusions and further work.

4.2 Rare Events in Regression

Our target applications are supervised prediction regression tasks guided by rare events - the occurrence of rare extreme values in a continuous target variable. These prediction tasks are often related to the anticipation of a critical phenomenon that is inherently continuous and for which an alarm may be triggered by a specific range of values of a continuous target variable. Thus, even though they refer to a more specific set of values, they constitute a particular case of regression problems where the utility of the predictions across the range of continuous target variable is not uniform. For this reason, utility-based regression is the proper evaluation framework for such applications.

In classification, event-driven prediction tasks are usually evaluated using the notions of precision and recall, which are preferred over other alternatives when in presence of large skew in the class distribution (Davis and Goadrich, 2006; Weiss, 2004). The main advantage of these statistics consists on their focus on the performance of the models on the events, completely ignoring their accurate predictions for the non-event class.

4.2.1 Relevance as a Continuous Notion of Event

The standard setup for event-driven classification is to have a so-called positive class that represents the target events while the negative class represents all the non-events. Proceeding in the same way in regression is difficult, given the infinite nature of the domain of a continuous target variable. On the other hand, the notion of relevance we have introduced and defined back in Chapter 3, Section 3.2.3 (see page 72), seems a natural way of expressing the preference biases of the user concerning a subset of values in a certain application. Moreover, this function allows the specification of different degrees of relevance with the already mentioned advantages in terms of our target applications that are eminently continuous.

As mentioned above, in classification the relevant events are the ones associated with a target class. This would correspond to establishing the cases belonging to this class as the only ones with any relevance. We can use our relevance function framework to express this domain-knowledge for classification tasks,

$$\phi(y) = I(y = C_E) \tag{4.1}$$

where I is the indicator function giving 1 if its argument is true and 0 otherwise, and C_E is the class describing the events (i.e. the positive class).

The information on the relevance function is domain-dependent. We have seen in Section 3.3.1 (see page 79) that defining this continuous function for regression domains might be difficult for the user. In Section 3.4.1 (page 96) we have described a method for specifying a continuous relevance function based on piecewise cubit Hermite interpolation. This method is based on the user-specification of a set of key points of the target relevance function. In the next section, we describe a heuristic method for automatically choosing these points in order to obtain a relevance function that gives higher relevance to rare extreme values of a continuous variable.

4.2.2 A Heuristic Relevance Definition for Rare Extreme Values

Our goal is to address applications where the rare events are described by the occurrence of not only rare but particularly extreme values on the continuous target variable, i.e. extreme low and/or high values. Moreover, in comparison to these values, the performance on the other more frequent values is basically irrelevant for the end user of these applications.

Regarding numeric variables, the rarity concept must be defined with respect to some distribution. It can be defined with respect to the sampling distribution, although ideally it should be defined with respect to the underlying (but typically unknown) distribution. Back in Chapter 2, and in

particular in Section 2.3.3 (page 47), we have mentioned some of the existing statistical methods for the identification of the so-called extreme values in the statistics field. Namely, we have referred to the *Extreme Value Theory* (Coles, 2001; Embrechts et al., 1997, 1998) employed for effective risk management on areas such as financial markets, but that has also shown to be useful in other fields like hydrology and climatology. Nevertheless, statistical methods such as this become very slow in the presence of big amounts of data. Alternatively, approximation techniques based on data samples like kernel density estimators (e.g. Hardle, 1990) can also be used. Nevertheless, and though some efforts have been made to improve the computational complexity of such density-based approximations (e.g. Yang et al., 2003), they are still computationally intensive and scalability remains difficult for practical applications.

As we have mentioned we are associating relevance with the rarity and extremeness of the values of the target variable. This means that if we assume a near normal distribution of the target variable, the relevance function is reasonably well approximated by the complement of the probability distribution function (pdf) of the target variable. In effect, our goal is to predict accurately the values on the tails of the distribution. In these conditions, our proposal is to use the box plot (Cleveland, 1993) (cf. Chapter 2, Section 2.3.3 - page 48), as a gross approximation of the distribution that nevertheless provides information on key points that can be used for our purposes.

Box plots provide a set of crucial statistics of the probability density function (pdf), and that are particularly interesting regarding extreme values. Namely, all values of the continuous variable above the high adjacent value given by $adj_H = Q_3 + 1.5\,IQR$, where Q_3 is the third quartile and $IQR = Q_3 - Q_1$ or below the low adjacent value, $adj_L = Q_1 - 1.5\,IQR$, are tagged as outliers and incorporate mild and extreme outliers. These two values define two sets of extreme outliers: the rare low extreme outliers (O_L) and the rare high extreme outliers (O_H).

Our objective is to obtain a continuous relevance function that maps the domain of the target variable Y to the interval $[0, 1]$, so that the extreme values of Y are assigned maximum relevance. In order to do so, we incorporate the adjacent values (adj_L and adj_H) as threshold values for the extremes and the median value ($\widetilde{Y}$) as a centrality value for irrelevance. We then use this set of control points, presented in Table 4.1, with the *piecewise cubic Hermite interpolation* method described in Section 3.4.1 (page 96) to obtain an extremes-based relevance function ϕ (cf. Figure 4.1).

The above derived relevance function ϕ fulfills the requirements of relevance for rare extreme values in general. Notice that we might have only one type of extremes, i.e. extreme high or extreme low outliers.

Having defined a heuristic method to obtain the relevance function we should remark that our methodology in no way depends on this method to define the function. This definition is useful

Table 4.1: Control relevance points for rare extreme values prediction.

Control points for ϕ	
y_k: box-plot statistic	$\phi(y_k)$
$adj_L = Q_1 - 1.5\,IQR$	1.0
$\widetilde{Y}$ = median of Y	0.0
$adj_H = Q_3 + 1.5\,IQR$	1.0

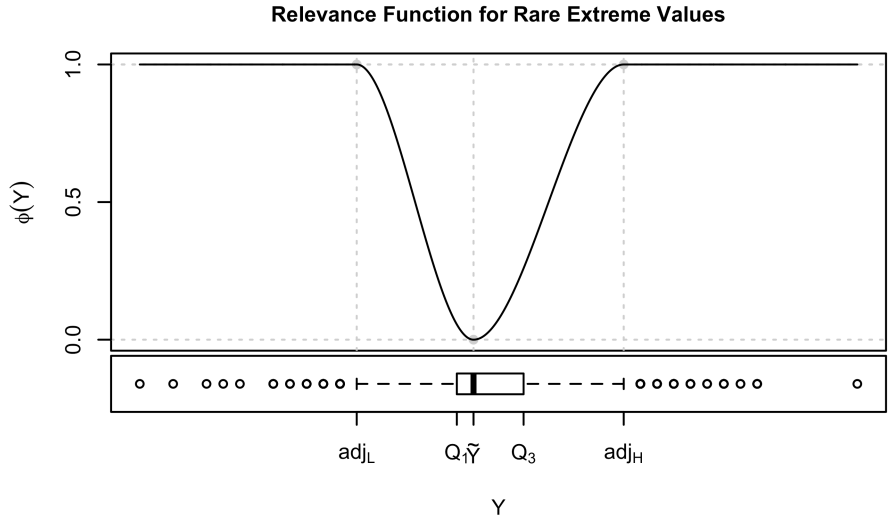


Figure 4.1: Our proposed extremes-based relevance function is generated from the box plot statistics for extreme values.

in cases where the user is unable to provide a formal definition of the notion of relevance for his application. There are many situations where the user simply has the notion that relevance is associated with extreme and rare values. In these cases, this heuristic method may help in defining a relevance function that is required for the application of our utility-based evaluation framework.

Having mentioned that events are a function of the user-assigned relevance, we now establish that relevant events are those for which $\phi(y) \geq t_E$, where t_E is a domain-dependent threshold on relevance.

Definition 4.1 (**Target Event**)**.** Let ϕ denote the relevance function of the target variable Y. We define the target event variable z by,

$$z := I(\phi(y) \geq t_E) \tag{4.2}$$

where I is the indicator function giving 1 if its argument is true and 0 otherwise, and $t_E \in [0, 1]$ is the pre-specified event threshold.

In classification, as relevance is usually a 0/1 function (c.f. Equation 4.1), this threshold is 1. For regression, in our default utility-based setting it would be 0, i.e. every value in the target variable is an event to some extent. In the context of the rare extreme values setting it is usually a good idea to focus the analysis of the performance of the models on a smaller subset of the values. This leads to setting the threshold to values near 1, depending what the user wants to consider as the target for the prediction task.

Example 4.1. Forest fires prediction

In Portugal, forests represent about 38% of its total land surface and their products are important contributors to the economic growth and prosperity. Wildfires are one of their main threats. These fires are very common, causing devastation and ruining vast areas of land, thus being a recurring problem in their preservation.

Forest fires may have various causes, such as human negligence and lightnings. Weather conditions are known to have an important part in the fire occurrence. Every year, the Portuguese government invests a large amount of money in fire prevention and fire fast detection.

Some solutions in fire detection (e.g. satellites, infrared scanners and other sensors designed to detect fire occurrences) are very expensive in terms of acquisition and maintenance. On the other hand, land organization and characteristics, air conditions, namely temperature and humidity, are known to affect fire occurrences and its evolution.

Bearing in mind that a better knowledge of these factors can lead to a better fire management, it would be interesting to have a model capable of supporting decisions, particularly in what concerns resources planning and firefighting. We have data of 2831 registered fires described by 86 attributes regarding land morphology (e.g. slope, elevation), land characteristics (e.g. coverage area of a tree type, road area), social-demographic values (e.g. active, retired, resident population) and weather variables.

The accurate prediction of large-scale forest fires would bring high benefits and avoid costs. Hence, from the available variables we selected the percentage of burnt area as the target variable. For the target event definition, we have used a value of the event threshold of one, i.e. $t_E = 1$ (cf. Definition 4.1). Applying this definition to the available data using the box-plot as the driver for obtaining the relevance function (shown in Figure 4.2), we may observe that the rare extreme values that form the target event are the fires where more than approximately 46% of area is burnt. This means that all the fires whose percentage of burnt area is above this value are considered to be rare and extreme values and target events for our prediction. It is at the prediction of these fires that we want the models to be particularly accurate.

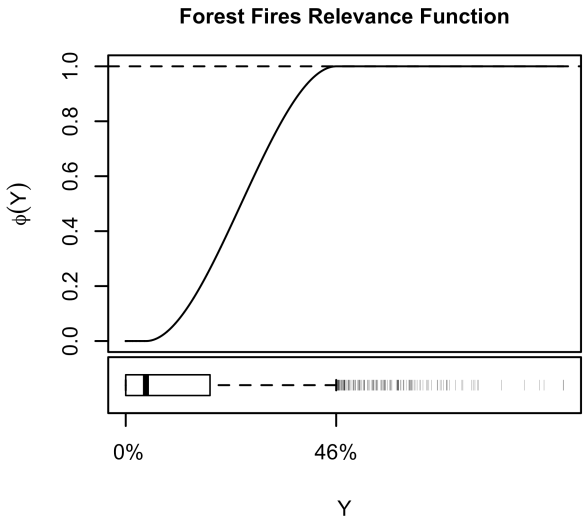


Figure 4.2: The target event for the Forest Fires problem is to anticipate the more extreme fires, i.e. the ones where more than 46% of the forest area is burnt.

4.3 Decision Process in Regression

In the context of event-driven classification problems, it is common to use confusion matrices for the characterization of the performance of a model. The numbers in this matrix (cf. Table 4.2) can be used to calculate several statistics used in classification with the objective of evaluating the model performance regarding the prediction of a target positive class. The positive class represents the rare events, while the negative class represents all non-events.

Table 4.2: The 2x2 confusion matrix.

Confusion Matrix			
	True		
Predicted	non-event	rare event	Total
no alarm	TN	FN	PNEG
alarm	FP	TP	PPOS
Total	NEG	POS	

Precision and recall are two metrics that can be obtained from a confusion matrix, which assesses the performance of the model based on its predictions over the event cases. Informally, precision measures the proportion of events signalled by the model that are real events, i.e. the ratio

TP/PPOS. recall measures the proportion of events occurring in the domain that are captured by the model, i.e. the ratio TP/POS.

Our goal is to be able to use these intuitive notions in the context of our applications, where the target variable is numeric. In order to be able to obtain such type of performance estimates, we need to find out when a regression model predicts an event. In the previous section, based on a relevance function ϕ and a threshold t_E, we have presented how any value of the continuous target variable Y is mapped to one of the two event/non-event classes. However, the same mapping function can not be applied to a prediction as a way to determine its class. In spite of $\phi(y) \geq t_E$ being the event condition for the true value y, $\phi(\hat{y}) \geq t_E$ is not the condition that determines the class of the prediction. The reason follows from the definition of the relevance and utility functions that we proposed in Chapter 3. For instance, in the example shown in Figure 4.1 (page 124), if we consider $t_E = 1$, any value y above adj_H is assigned the event class. If a prediction $\hat{y}$ for y is below adj_L, given the respective value of $\phi(\hat{y})$ it would also be considered an event and, thus a correct prediction of y. However, although both y and $\hat{y}$ refer to rare values, they are different events: one an unusually low value while the other is an extreme high value. Moreover, any other prediction close to y but with $\phi(\hat{y}) < 1$, would be tagged as non-event.

As an alternative, we propose the use of the utility of each prediction. The reasoning is based on the relationship between the utility function and an event. The target event z is defined based on the relevance function ϕ and on a threshold value t_E (cf. Definition 4.1). According to the limits of utility of a prediction, proven by Theorem 3.21 on Chapter 3 (page 92), we know that: (i) the maximum of $U^p_\phi(\hat{y}, y)$ is $\phi(y)$; (ii) the minimum of $U^p_\phi(\hat{y}, y)$ is $-\phi^p(\hat{y}, y)$. In this sense, the higher the precision of the predictions the closer is U^p_ϕ to $\phi(y)$. This means that utility can be regarded as a function that is closely related to the probability estimate of the prediction $\hat{y}$ being in the same class of y.

Any increasing or decreasing function of class membership probability is sufficient to induce a correct ranking of examples (Zadrozny and Elkan, 2002). In this context, we have decided to use the ranking information given by the utility function as scores. In other words, we take the utility calculated for a given prediction as the ranking position of the instance i for the target event membership. To achieve that, we rescale our utility values from the $[-1, 1]$ interval into a $[0, 1]$ interval of event membership. However, such scaling is not enough. As referred by Zadrozny and Elkan, in cost-sensitive domains it is important to have an accurate estimate of the probability that each test example is a member of the event class. The reason is that each prediction, and thus each decision, carries an additional cost/benefit. In order to obtain accurate probability estimates, we need to calibrate the utility ranking scores. Through calibration, we get scores that can be directly interpreted as the chances of membership to the target class.

As calibration method (e.g. Zadrozny and Elkan, 2001, 2002), we have chosen the *Pool Adjacent Violators* (PAV) method (Fawcett and Niculescu-Mizil, 2007). The PAV algorithm, as we have previously mentioned on Chapter 2, Section 2.3.2 (page 30), is a calibration technique that finds an isotonic (increasing monotonic) transformation of the scores that minimizes the Brier Score (BS). The Brier Score, or in other words, the mean squared error for probabilistic classifiers, is defined as,

$$BS = \frac{1}{n}\sum_{i=1}^{n}(s_i - z_i)^2 \tag{4.3}$$

where s_i is the calibrated score and z_i is the non-event/event class.

Algorithm 4.1 ubaScores($\mathbf{u}, \mathbf{z}$): Generation of utility-based calibrated scores based on Fawcett and Niculescu-Mizil (2007).

Input: set of raw utility scores $\mathbf{u} = \{u_i\}_{i=1}^{n} \in [-1, 1]$;
for set of true event values $\mathbf{z} = \{z_i\}_{i=1}^{n} \in \{0, 1\}$.
Output: **s**: calibrated utility-based scores.

```
o ← order(u, decreasing = false)
s ← sort(z, order = o)                          // update true events by the scores order
repeat
    i ← n − 1
    pooled ← false
    while i > 0 do
        (k, i) ← maxNonIncrPath(s)
        if i ≠ k then
            s_{k:i} ← Σ_{l=k}^{i} s_l/(k − i + 1)
            pooled ← true
        end if
        i ← k − 1
    end while
until !pooled
return s
```

After calibrating the ranking utility scores by the *pool adjacent violators* algorithm, we get a new set of scores that besides providing an accurate ranking, are also a more reliable set of probability estimates. These new scores give us utility-based probability estimates of extreme values for the predictions of regression models. Next, we exemplify on the forest fires domain how we obtain probability estimates from the raw utility scores of a set of predictions.

Example 4.2. From raw utility scores to probability estimates: forest fires example.

Suppose that we have in the forest fires domain a test set with 15 instances. Table 4.4 shows the predicted values ($\hat{y}_i$) of some model, lets say M_1*, for the true values (y_i), the respective utility raw*

scores (u_i) and the associated true class (z_i) regards extremeness. Given this input, how does the above algorithm ubaScores *proceeds to obtain a set probability estimates for the membership to the positive class? The set of calibrated scores is initialized with the true classes z_i, which assigns a probability of 1 to the positive examples and 0 to the negative examples. These scores are first sorted by the raw utility, such that $u_0 \geq u_1 \geq \cdots \geq u_{14}$. The algorithm will then look iteratively for the maximum length path of* adjacent violators, *i.e. the maximum number of non increasing scores. When it finds any set of instances on these conditions, it replaces their scores by the overall average.*

In this particular case, the first set of instances with non-increasing scores is 12-10. It then replaces the scores of these 3 instances by 1/3 and proceeds to instance number 9, until it reaches instance 0. It goes back to the beginning and the process is repeated until no more adjacent violators *are found across the scores.*

Table 4.3: Raw utility scores vs utility-based probability estimates.

Calibration of raw utility scores									
				Probabilities Estimation					
#	$\hat{y}_i$	y_i	u_i	z_i					s_i
0	74.2	72.6	0.96404	1	1	1	1	1	1
1	49.5	43.6	0.85436	0	0	0	1/7	2/9	1/4
2	60.7	44.1	0.61908	0	0	0	1/7	2/9	1/4
3	52.4	32.3	0.37382	0	0	0	1/7	2/9	1/4
4	52.4	30.5	0.30166	0	0	0	1/7	2/9	1/4
5	59.6	29.7	0.15079	0	0	0	1/7	2/9	1/4
6	2.5	0.1	0.00000	0	0	0	1/7	2/9	1/4
7	7.3	50.6	−0.14404	1	1	1	1/7	2/9	1/4
8	51.4	14.3	−0.15856	0	0	1/2	1/2	2/9	1/4
9	2.9	47.1	−0.23515	1	1	1/2	1/2	2/9	1/4
10	60.2	6.8	−0.30155	0	1/3	1/3	1/3	1/3	1/4
11	74.5	18.6	−0.30278	0	1/3	1/3	1/3	1/3	1/4
12	2.1	58.2	−0.31814	1	1/3	1/3	1/3	1/3	1/4
13	49.5	0.0	−0.50000	0	0	0	0	0	0
14	75.2	0.0	−0.50000	0	0	0	0	0	0

The above derived sequence of calibrated scores s_i is used as probability estimates of membership to the positive class.

Even though Algorithm 4.1 enables us to convert utility values into probability estimates of the positive class, one must stress that in this process some information is necessarily lost. Namely, the typical sensitivity of regression domains is absent as the next example illustrates.

Example 4.3. Two different models with the same calibration: forest fires example.

Based on model M_1*, we can derive a second model* M_2 *that makes different predictions for all the values and still achieves the same calibrated scores as* M_1*. Table 4.4b shows such model and the respective scores involved on this small illustrative domain.*

Table 4.4: Two different models with equal utility-based probability estimates.

(a) Model M1

Calibration of raw utility scores					
#	$\hat{y}_i$	y_i	u_i	z_i	s_i
0	74.2	72.6	0.96404	1	1.00
1	49.5	43.6	0.85436	0	0.25
2	60.7	44.1	0.61908	0	0.25
3	52.4	32.3	0.37382	0	0.25
4	52.4	30.5	0.30166	0	0.25
5	59.6	29.7	0.15079	0	0.25
6	2.5	0.1	0.00000	0	0.25
7	7.3	50.6	−0.14404	1	0.25
8	51.4	14.3	−0.15856	0	0.25
9	2.9	47.1	−0.23515	1	0.25
10	60.2	6.8	−0.30155	0	0.25
11	74.5	18.6	−0.30278	0	0.25
12	2.1	58.2	−0.31814	1	0.25
13	49.5	0.0	−0.50000	0	0.00
14	75.2	0.0	−0.50000	0	0.00

(b) Model M2

Calibration of raw utility scores					
#	$\hat{y}_i$	y_i	u_i	z_i	s_i
0	72.8	72.6	0.99640	1	1.00
1	46.6	44.1	0.93728	0	0.25
2	46.4	43.6	0.92434	0	0.25
3	46.5	32.3	0.48019	0	0.25
4	47.0	30.5	0.39371	0	0.25
5	26.1	47.1	0.35101	1	0.25
6	48.1	29.7	0.34380	0	0.25
7	0.3	0.1	0.00000	0	0.25
8	11.8	50.6	−0.04043	1	0.25
9	50.4	18.6	−0.05694	0	0.25
10	47.5	14.3	−0.12766	0	0.25
11	47.2	6.8	−0.22561	0	0.25
12	2.7	58.2	−0.30719	1	0.25
13	46.7	0.0	−0.50000	0	0.00
14	49.8	0.0	−0.50000	0	0.00

In order to distinguish between these two models we will need to take full advantage of the utility information.

Given a ranked list of scored examples, it is necessary to define a cut-off threshold t so that all examples that are ranked higher than t are classified as positive and the remaining as negative. Informally, the choice of the cutoff t should be done in such a way that the chance of all the instances belonging to the positive class is higher than the chance of belonging to the negative class, considering both the costs of false positives and of false negatives.

For probabilistic classifiers in uniform cost domains (i.e. where false positives and false negatives have the same cost), the best cutoff t is the point where the probability of being positive is higher than the probability of being negative, that is $t > 0.5$. This means that if the probability estimate

$s_i = \hat{P}(1|\mathrm{x}_i) \geq 0.5$ then instance x_i is assigned the positive class ($\hat{z}_i = 1$) otherwise it is assigned the negative class ($\hat{z}_i = 0$).

This framework assumes that the domain is static, i.e. the distribution of the training data is precise and not expectable to change. This is a very strong assumption and not applicable to real-world domains, which are characterized by being imprecise regards the class distribution and/or error costs. Moreover, in our target applications the cost of an error or the benefit of a hit is not constant throughout the domain. In a non-uniform utility scenario such as ours, where the costs/benefits are example-dependent, the optimal decision threshold varies with the examples.

As mentioned by Elkan (2001) defining a benefit matrix is, most of the times, more intuitive than using cost matrices. From this point of view, the predictions of a model are evaluated from the perspective of their benefits, with incorrect predictions being assigned negative benefits while correct predictions get positive scores. In Chapter 3 we defined the utility function following this same idea. In this context, to use a benefit matrix, and we must ensure that utility values (i.e. costs or benefits) are assigned to all entries of the matrix.

Our proposal is to define a benefit matrix $B(\hat{z}, z, u)$ where each entry is described by a predicted class $\hat{z}$ [*1], a true class z and a raw utility score u of the prediction. The assignment of costs and benefits in a benefit matrix obeys two rules: (i) if the prediction represents a true prediction of the negative class ($B(0, 0, u)$ or TN) or a true prediction of the positive class ($B(1, 1, u)$ or TP) then the benefits should be non-negative; (ii) if the prediction represents a false prediction of the negative class ($B(1, 0, u)$ or FN) or a false prediction of the positive class ($B(0, 1, u)$ or FP) then the benefits should be negative, i.e. costs.

In the case of TP we want the benefit to be proportional to the utility of the prediction. We normalize the utility values to the interval $[0, 1]$ using the increasing function $(1+u)/2$. In the case of FP we want the cost to be inversely proportional to the utility. In this context, we normalize the utility values to the interval $[-1, 0]$ with the increasing function $-(1-u)/2$.

Table 4.5 presents our proposal concerning the benefit matrix $B(\hat{z}, z, u)$ representing the benefit of predicting class $\hat{z}$ for class z given the raw utility score $u = U^p_\phi(\hat{y}, y)$.

In the context of event-based prediction problems most of the performance metrics, such as precision and recall, are relative to the set of positive predictions made by the models. In this sense, we need to transform the above benefit matrix into a cost-benefit matrix where the costs or benefits are relative only to positive predictions. For this transformation we used the technique mentioned in Fawcett (2006b). We eliminate the negative predictions and incorporate their costs

[*1] We denote the class non-event by 0 and the class event by 1.

Table 4.5: The benefit matrix $B(\hat{z}, z, u)$ for the prediction of rare extreme values events based on utility.

Benefit Matrix $B(\hat{z}, z, u)$		
	True (z)	
Predicted $(\hat{z})$	non-event	rare event
no alarm	$(1-u)/2$	$-(1+u)/2$
alarm	$-(1-u)/2$	$(1+u)/2$

or benefits into the positive predictions row by (i) subtracting the first row from both rows of the benefit matrix; and then (ii) change the sign of the cell corresponding to FP, so it becomes a cost. Following this procedure, and considering the benefit matrix $B(\hat{z}, z, u)$ presented in Table 4.5, we obtain the so-called cost-benefit matrix $B_1(\hat{z}, z, u)$ shown in Table 4.6.

Table 4.6: The transformed cost-benefit matrix $B_1(\hat{z}, z, u)$ for the prediction of rare extreme values events.

Cost-Benefit Matrix $B_1(\hat{z}, z, u)$		
	True (z)	
Predicted $(\hat{z})$	non-event	rare event
no alarm	0	0
alarm	$1-u$	$1+u$

In the above setting, similarly to what was enunciated by Elkan, the optimal predicted class for an example y_i is determined by the maximization of the utility score, which is given by

$$\hat{z}_i = \underset{\hat{z}_i \in \{0,1\}}{\operatorname{argmax}} \sum_{z_j \in \{0,1\}} \hat{P}(z_j|\mathrm{x})\, B_1(\hat{z}_i, z_j, u_i) \tag{4.4}$$

where $B_1(\hat{z}_i, z_j, u_i)$ is the benefit of predicting class $\hat{z}_i$ for a true class z_j, with the associated raw utility score u_i.

From Equation 4.4 we obtain,

$$\hat{z}_i = \underset{\hat{z}_i \in \{0,1\}}{\operatorname{argmax}} \{(1-s_i)\,(1-u_i), s_i\,(1+u_i)\} \tag{4.5}$$

where s_i is the calibrated score that provides a reliable probability estimate for the membership of $\hat{y}_i$ to the positive class $\hat{P}(1|\mathrm{x})$.

Using this technique we are now able, for each example y_i, to determine the optimal decision threshold to assign the prediction $\hat{y}_i$ to the positive class. The predicted class for a given instance i is determined by the following function,

$$\hat{z}_i = I(s_i\,(1+u_i) > (1-s_i)\,(1-u_i)) \tag{4.6}$$

which is equivalent to

$$\hat{z}_i = I\left(s_i > \frac{1-u_i}{2}\right) \tag{4.7}$$

where s_i is the calibrated utility score, u_i is the raw utility score and I is the indicator function giving 1 if its argument is true and 0 otherwise.

4.3.1 Precision and Recall with Instance Varying Utility

Precision and recall are two of the most commonly used metrics to estimate the performance of models in event-driven and highly skewed domains (Davis and Goadrich, 2006), such as our target domains. The main advantage of these statistics is that they are focused on the performance on the events, disregarding predictions made for the non-event class. There is a known trade-off between these two statistics (always outputting an event signal yields to 100% recall but with a very poor precision as most signals will be wrong), and often the two are put together in a single weighted score like for instance the F-measure (Rijsbergen, 1979).

One of our proposals in this chapter is to provide the equivalents of these two statistics for regression problems in order to properly evaluate the performance of the models on the values that really matter. As we are dealing with applications with non-uniform costs / benefits of the target variable, the proportions provided by precision and recall are now expressed in terms of the utility obtained by the model instead of a hit/miss ratio as in classification. In practice, each instance after being mapped in the confusion matrix is replicated according to its benefit or cost given by the benefit matrix of Table 4.6.

Before we present the formal definitions of recall and precision for numeric prediction problems, we need to define two auxiliary concepts: the total benefits obtained by classifying as positive all the event examples (Pbenef), and the total costs obtained by classifying as positive all the non-event examples (Ncost).

From the Theorem 3.21 in Chapter 3, we know that the maximum utility of an event is achieved when the prediction equals the true value and it is equal to $\phi(y_i)$. Hence, and according to the cost-benefit matrix presented in Table 4.6, Pbenef is given by

$$\text{Pbenef} = \sum_{i:z_i=1} 1 + \phi(y_i) \tag{4.8}$$

Similarly, by Theorem 3.21 in Chapter 3, we also know that the minimum utility, i.e. maximum cost, of a non-event is achieved when the prediction belongs to a relevance bump different from the

true value and its relevance is maximum, i.e. $\phi(\hat{y}_i) = 1$. Hence, and according to the cost-benefit matrix presented in Table 4.6, Ncost is given by

$$\text{Ncost} = \sum_{i:z_i=0} 2 - p\,(1 - \phi(y_i)) \tag{4.9}$$

In this context, we propose the following definitions for precision and recall.

Definition 4.2 (**Recall**)**.** Recall is the proportion of utility obtained by the positive predictions for the events over the total utility of these events.

$$recall = \frac{\sum\limits_{i:\hat{z}_i=1,z_i=1} (1 + u_i)}{\sum\limits_{i:z_i=1} (1 + \phi(y_i))} \tag{4.10}$$

where z_i is the true class, $\hat{z}_i$ is the predicted class, u_i is the raw utility score and $\phi(y_i)$ is the relevance of true value y_i.

Definition 4.3 (**Precision**)**.** Precision is the proportion of utility obtained by the positive predictions for the events over the total utility of cases predicted as positive.

$$precision = \frac{\sum\limits_{i:\hat{z}_i=1,z_i=1} (1 + u_i)}{\sum\limits_{i:\hat{z}_i=1,z_i=1} (1 + \phi(y_i)) + \sum\limits_{i:\hat{z}_i=1,z_i=0} (2 - p\,(1 - \phi(y_i)))} \tag{4.11}$$

where z_i is the true class, $\hat{z}_i$ is the predicted class, u_i is the raw utility score, $\phi(y_i)$ is the relevance of true value y_i, and p is the cost penalization factor.

The example below illustrates how these two new definitions of precision and recall meet the requirements of our target applications.

Example 4.4. Precision and recall in regression: forest fires example.

Let us consider the models M_1 *and* M_2 *presented in Example 4.3. We have 15 test instances from which only 4 are positive according to the rare extreme values event definition given in Example 4.1.*

To estimate the precision and recall of both models, we need to obtain the predicted class by each model for the test cases. Using the benefit maximization criterion presented in Equation 4.4,

Table 4.7: Establishing decision thresholds based on utility.

(a) Model M1

Decision Thresholds based on Utility						
#	$\hat{y}_i$	y_i	s_i	u_i	$\hat{z}_i$	z_i
0	74.2	72.6	1.00	0.96404	1	1
1	49.5	43.6	0.25	0.85436	1	0
2	60.7	44.1	0.25	0.61908	1	0
3	52.4	32.3	0.25	0.37382	0	0
4	52.4	30.5	0.25	0.30166	0	0
5	59.6	29.7	0.25	0.15079	0	0
6	2.5	0.1	0.25	0.00000	0	0
7	7.3	50.6	0.25	−0.14404	0	1
8	51.4	14.3	0.25	−0.15856	0	0
9	2.9	47.1	0.25	−0.23515	0	1
10	60.2	6.8	0.25	−0.30155	0	0
11	74.5	18.6	0.25	−0.30278	0	0
12	2.1	58.2	0.25	−0.31814	0	1
13	49.5	0.0	0.00	−0.50000	0	0
14	75.2	0.0	0.00	−0.50000	0	0

(b) Model M2

Decision Thresholds based on Utility						
#	$\hat{y}_i$	y_i	s_i	u_i	$\hat{z}_i$	z_i
0	72.8	72.6	1.00	0.99640	1	1
1	46.6	44.1	0.25	0.93728	1	0
2	46.4	43.6	0.25	0.92434	1	0
3	46.5	32.3	0.25	0.48019	0	0
4	47.0	30.5	0.25	0.39371	0	0
5	26.1	47.1	0.25	0.35101	0	1
6	48.1	29.7	0.25	0.34380	0	0
7	0.3	0.1	0.25	0.00000	0	0
8	11.8	50.6	0.25	−0.04043	0	1
9	50.4	18.6	0.25	−0.05694	0	0
10	47.5	14.3	0.25	−0.12766	0	0
11	47.2	6.8	0.25	−0.22561	0	0
12	2.7	58.2	0.25	−0.30719	0	1
13	46.7	0.0	0.00	−0.50000	0	0
14	49.8	0.0	0.00	−0.50000	0	0

we can find for each instance the best decision threshold and assign the appropriate class. The predicted classes are shown in Tables 4.7a and 4.7b .

For comparison purposes we start by calculating the values of precision and recall of M_1 *and* M_2*, using the standard classification-based formulas, using the classes predicted by the models. We present these scores together with the respective F-measure (Fm) on Table 4.8, with different values for importance of recall relative to precision - equal importance with* $\beta = 1$*, twice of the importance with* $\beta = 0.5$ *and half of the importance with* $\beta = 2$*. This table also shows the Mean Absolute Deviation (*MAD*) and Mean Utility (*MU*) of the numeric predictions of the models.*

Table 4.8: Precision and Recall without utility information.

Estimated Performance							
	MAD	MU	precision	recall	Fm	$\text{Fm}_{\beta=0.5}$	$\text{Fm}_{\beta=2}$
M_2	24.802	0.178	0.333	0.250	0.286	0.313	0.263
M_1	34.205	0.054	0.333	0.250	0.286	0.313	0.263

We can confirm that using these standard definitions of precision, recall and F-measure, the two models have the same scores. However, as we have previously mentioned and as it can be confirmed by looking at the values of MU*, model* M_2 *is an improved version of* M_1*, particularly on observations with rare extreme values, and thus should get better scores.*

Table 4.9 presents the same results when we use our proposed utility-based notions of precision and recall. These new metrics clearly identify the difference between the models regarding the numeric prediction of the target event.

Table 4.9: Precision and Recall with utility information.

	Estimated Performance						
	MAD	MU	precision	recall	Fm	$\text{Fm}_{\beta=0.5}$	$\text{Fm}_{\beta=2}$
M_2	24.802	0.178	0.333	0.248	0.285	0.312	0.262
M_1	34.205	0.054	0.328	0.246	0.281	0.308	0.259

Through the use of utility in our proposed definitions of precision and recall we are able to penalize not only *false alarms* (FP) and *missed opportunities* (FN), but also *confusing events*. In an application where both type of extremes, high and low, are of interest of the end user, a *confusing event* should be heavily penalized. In the stock market domain, for instance, if a model advises the user to buy when in fact should sell, it is a serious error. By using utility, that same prediction would be considered as a cost (i.e. negative utility value). Moreover, the amount of penalization is more sensitive, in the sense that it takes on a continuous scale depending on the difference between the true and predicted values because of the definition of utility.

4.4 Ranking Analysis in Regression

Performance metrics like precision and recall tell us how good is a model regards predicting a certain target event. The predictions of these models are cast into an event / non-event class by applying a decision threshold t over a score function. In the previous section we have proposed a solution that assigns each prediction to the event / non-event class based on its utility. However, using a fixed decision threshold is not always considered as the most suitable way of comparing models. On most cost-sensitive and non-uniform domains such as our target applications, it is difficult to known and to pre-establish a reliable threshold value for all the cases. In these domains an overall evaluation of the models over different operating conditions (e.g. different threshold values) is usually considered a more trustful analysis. Moreover, the event / non-event class is defined based on the relevance of the target continuous variable. Although we are interested in event-based predictions, relevance is a continuous concept rather than binary. The same applies to the raw and calibrated utility scores obtained for the predictions. The use of a decision threshold creates crisp and artificial divisions that go against these continuity properties of our target domains.

In this context, a ranking analysis of the model is more suitable. This analysis is not based on the binary predictions made by the model, but on the ranked set of examples provided by the scoring function, before applying a decision threshold. Receiver Operating Characteristic (ROC)

analysis (e.g. Fawcett, 2006a), as we have mentioned in Chapter 2 (Section 2.3.2 - page 22), allows a comparison of the models independently of the decision threshold. A ROC curve plots the true positive rate (TPR) and the false positive rate (FPR) obtained for each possible decision threshold. This means that the ROC curve obtained for a model is the same regardless of the operating conditions (class skew or error costs). For these reasons, ROC analysis is one of the most used techniques in problems with skewed distributions or cost-sensitive domains. Still, whenever the proportion of positive instances is too small, such as the prediction of rare values, Precision-Recall (PR) curves are preferred over ROC curves. PR curves give us a more realistic perspective of the model performance over a skew distributed data set (Davis and Goadrich, 2006).

PR curves have been widely used (e.g. Brodersen et al., 2010; Clémençon and Vayatis, 2009; Davis and Goadrich, 2006) as a tool to visualize and evaluate the performance of models with respect to their discriminating abilities between two classes. By comparing models using PR analysis we are inspecting how well the scoring function ranks the examples. If the scoring function is a good ranker then this means that it ranks all the positive (target) examples first and the negative examples afterwards. This also means that independently of the decision threshold the model is able to discriminate the positive class from the negative class.

Davis and Goadrich proved that for a fixed number of positive and negative examples there is an one-to-one correspondence between a curve in the ROC space and a curve in the PR space, such that the curves contain exactly the same confusion matrices, if recall is different from zero. That correspondence is established in the PN space (Flach, 2007) where the information on TP and FP is gathered. This is an unscaled space where a point $\langle \mathrm{TP}, \mathrm{FP} \rangle \in [0, \mathrm{POS}] \times [0, \mathrm{NEG}]$ represents, for a given number of positive examples (POS) and negative examples (NEG), the set of positive predictions (TP and FP).

Once defined, a PN graph can be transformed into a ROC or PR curve. Both ROC and PR curves depend on the same ground information: the true positives (TP) and the false positives (FP) at each point. From this information, what ROC and PR curves do is to illustrate different insights about the model behavior in the space $[0, 1] \times [0, 1]$, through the calculation of different statistics.

ROC curves plot the false positive rate in X-axis and the true positive rate in Y-axis. Hence, to convert points of PN space into ROC space points, the following mapping is applied

$$\langle \mathrm{TP}, \mathrm{FP} \rangle \mapsto \left\langle \frac{\mathrm{FP}}{\mathrm{NEG}}, \frac{\mathrm{TP}}{\mathrm{POS}} \right\rangle \tag{4.12}$$

where POS is the total number of positive examples and NEG is the total number of negative examples.

PR curves plot the true positive rate, i.e. recall, in X-axis and the precision in the Y-axis. Hence, to convert points of PN space into PR space points, the following mapping is applied

$$\langle \mathrm{TP}, \mathrm{FP} \rangle \mapsto \left\langle \frac{\mathrm{TP}}{\mathrm{POS}}, \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \right\rangle \tag{4.13}$$

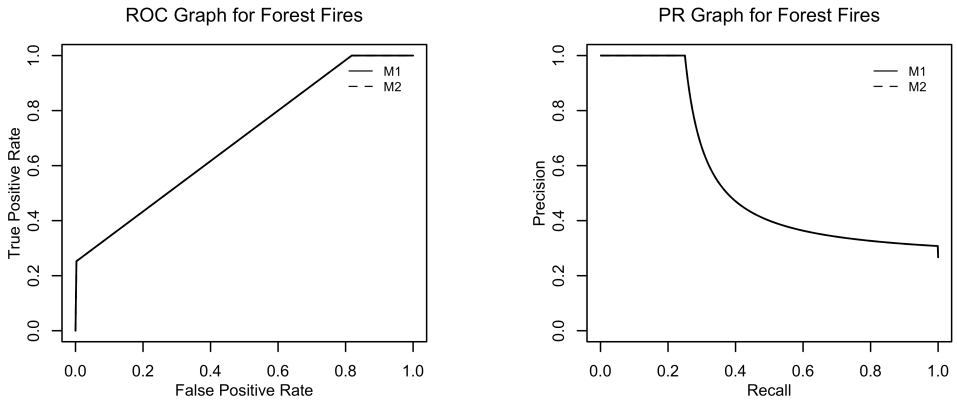
where POS is the total number of positive examples.

Still, while in the ROC space the interpolation of points is linear, in the PR space it is not. As we have discussed in Chapter 2, Davis and Goadrich (2006) have shown the erroneous effect that a wrong interpolation in PR space can have in the comparison of models.

In the following example, we generate a PN graph and apply these mapping functions to obtain the correspondent ROC and PR curves.

Example 4.5. ROC and Precision-Recall curves: forest fires example.

We return to the models M_1 *and* M_2 *that we have been working on. They obtain the same calibrated scores* $\mathbf{s}$ *for the test cases (cf. Example 4.3), and thus rank them in the same way. As it is possible to observe in the graphs on Figure 4.3 their ROC and PR curves are equivalent. Still, as we known these are two different models. The identical curves are a result of not including the utility information on the ranking analysis.*



(a) (b)

Figure 4.3: Forest Fires: ROC and PR curves obtained by different models.

From the above example, we see that, in the current setup, ranking analysis can yield to misleading conclusions. Similarly to what we have done with precision and recall definitions, we need to incorporate the utility information into the PN space. In the next section, we explain how to generate a PN space sensitive to utility and, in that space, derive Precision-Recall curves.

4.4.1 Precision-Recall Curves with Instance Varying Utility

On certain real-world applications costs may vary with examples (e.g. credit fraud, target marketing). Fawcett (2006b) proposed a technique to incorporate instance-varying costs into what he called ROC instance-varying curves (ROCIV). The objective was to address applications where the benefits across the positive examples and the costs across negative examples are not necessarily constant. The ROCIV curves are plotted in a space where the X-axis represents the fraction of the total FP cost, and the Y-axis represents the fraction of total TP benefits. In this scenario, each instance is replicated in proportion to its cost or benefit. The same technique is employed here to incorporate utility into the ROC and PR curves. Therefore, to produce such curves, we generate through the ubaPN algorithm (cf. Algorithm 4.2) a PN graph with instance varying utilities. The main particularity of this algorithm is that instead of incrementing TP and FP instance counts, as in standard classification, it increments the TP utility fraction (TPbenef) and the FP utility fraction (FPcost) according to each instance. The final PNutil points correspond to utilities fractions in the utility space associated to positive and negative examples

Algorithm 4.2 ubaPN($\mathbf{s}, \mathbf{z}, \mathbf{u}$): Generation of PNutil points with instance varying utilities.

Input: set of calibrated scores $\mathbf{s} = \{s_i\}_{i=1}^{n} \in [0, 1]$;
set of true event values $\mathbf{z} = \{z_i\}_{i=1}^{n} \in \{0, 1\}$;
set of raw utility scores $\mathbf{u} = \{u_i\}_{i=1}^{n} \in [-1, 1]$;
Output: PNutil: PNutil points with instance varying utility by increasing TPbenef.

PNutil $\leftarrow \langle\rangle$ // initialization
$s_{prev} \leftarrow -\infty$
for $i \leftarrow 1$ **to** n **do**
  if $s_i \neq s_{prev}$ **then**
    PNutil $\leftarrow$ push($\langle$TPbenef, FPcost$\rangle$, PNutil)
    $s_{prev} \leftarrow s_i$
  end if
  if $z_i = 1$ **then**
    TPbenef $\leftarrow$ TPbenef $+ (1 + u_i)$ // is a positive example
  else
    FPcost $\leftarrow$ FPcost $+ (1 - u_i)$ // is a negative example
  end if
end for
PNutil $\leftarrow$ push($\langle$TPbenef, FPcost$\rangle$, PNutil)
return PNutil

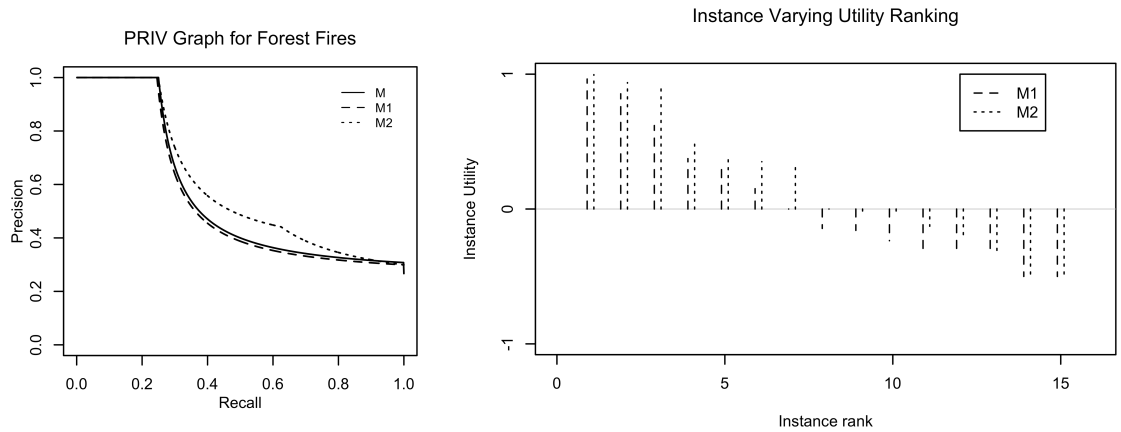
Based on the PNutil space, ROC and PR curves with instance varying utilities can be drawn. Given that our target events are rare, we focus on the analysis of PR curves by their known

effectiveness in this type of prediction tasks (Davis and Goadrich, 2006). Next, we exemplify how the incorporation of utility affects a PR curve.

Example 4.6. A Precision-Recall curve: forest fires prediction.

Taking again the domain of Forest Fires, we have produced once more a PR curve to each one of the models M_1 *and* M_2*, but now taking utility into account. The result is on Figure 4.4a. Comparatively to the previously obtained PR curves, which we here identify as* M *(cf. Figure 4.4a), the two models achieve different curves. In particular, we have that the new curves have an endpoint different from the previous one, where no utility values were considered. The reasoning is that the generality measure achieved for maximum recall was* POS/(POS+NEG) *for* M *and is* Pbenef/(Pbenef+Ncost) *for the new curves, which in this case do not correspond to same ratio. From an overall perspective and comparatively to the previous curve* M*, we see that the curve* M_1 *shows to be slightly worse, while the curve of* M_2 *indicates that this model is able to achieve a better tradeoff between precision and recall.*

In the following Figure 4.4b, it is possible to have a closer look at the way each of the models ranks its instances and its respective utility value. Ideally, all the positive instances should be ranked first, followed by all negative instances, and by decreasing value of utility. From this perspective, M_2 *is the model that distinguishes better the positive examples (i.e. event instances) from the negative examples (i.e. non-event instances).*



(a) (b)

Figure 4.4: Forest Fires: PR curves with instance varying utility obtained by different models.

In summary, the PR curves with instance varying utilities provide us a number of interesting insights for model evaluation and analysis since it shows the variability of the ranking performance for different prediction regions.

4.4.2 Utility Ranking Metrics

Even though PRIV curves can be very informative, we need summary measures of these curves, so that a more formal statistical analysis can be performed. Our goal is to obtain an approximation of the Area Under Curve for our proposed PRIV curves (AUC − PRIV). In Chapter 2, Section 2.3.2.4, we have presented some performance metrics that are based on standard PR curves. Namely, AUC − PR (Davis and Goadrich, 2006), which is the empirical approximation of to Area Under Curve for PR Curve, MAP11 (Manning et al., 2008), which is the Mean Average Precision obtained for a set of 11 pre-specified levels of recall, BFM (Altidor et al., 2009), which is the Best F-measure obtained within all the PR curve points. All these metrics were originally proposed in the context of classification. Their calculation involve concepts of standard PR curves. Our objective is to use PR curves with instance varying utilities (PRIV) in their evaluation.

Fawcett (2006b) proved that Area Under Curve for the ROCIV curves (AUCIV) is equivalent to the probability that a classifier will rank a random positive instance higher than a random negative instance, given that the examples are chosen in proportion to their cost. With this proof, Fawcett established the equivalence between the area under ROC curves (AUC) and the area under ROCIV curves (AUCIV) with the stipulation that the instances are selected according to their costs.

We have used this equivalence to approximate the Area Under Curve for our PRIV curves (AUC − PRIV). Assuming that instances are chosen in proportion to their utility, we compute the achievable PRIV curve and then by the composite trapezoidal rule obtain an approximation of the Area Under Curve for PRIV curves (AUC-PRIV). This is the same methodology previously described in Chapter 2, Section 2.3.2.3 (see page 38). The difference is that here we use an instance varying utility space. To calculate the AUC-PRIV, it is important as mentioned by Davis and Goadrich (2006) for the standard PR curves, to first standardize our PRIV curves to always cover the full range of recall values [0,1] and then interpolate between points. As with ROCIV curves, the optimal AUC − PRIV is 1, however the AUC − PRIV for a random classifier is equal to Pbenef/(Pbenef + Ncost), as this is the expected precision for classifying a random sample of examples as positive.

In the following example, we have performed macro-averaging of our results to calculate the AUC − PRIV, where the AUC − PRIV is first calculated for each fold and then averaged to produce one value. MAP11 can be seen as a less accurate estimate of AUC-PRIV (Goadrich et al., 2006).

Still, it is more efficient as it requires the calculation and interpolation of points on the curve for a limited number of recall levels. As we will see, BFm can give an optimistic estimate of the model's performance. Hence, it will not act as good estimator for all levels of recall.

Example 4.7. Utility ranking performance: forest fires prediction.

With the objective of making a utility-based ranking analysis, we have run 3 different models, A, B *and* C*, on the forest fires domain through 1x10 stratified cross-validation. The set of PRIV curves produced by the three models is presented in Figure 4.5.*

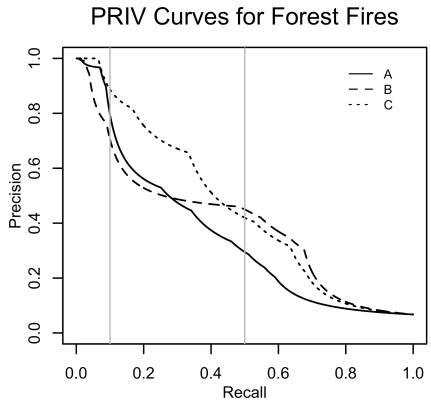


Figure 4.5: Forest Fires: PR curves obtained by different models.

From a first visual inspection, we can say that model C *is the best ranker. Still, we may be interested in knowing which is the best ranker, considering that we accept no less than a certain level of recall (10% or 50%). We might as well be interested to know if the conclusions achieved by* AUC − PRIV *are different from the ones achieved by two other estimators, namely* Mean Average Precision - MAP11 *and* Best F-measure - BFM*. Table 4.10 presents the obtained results.*

Table 4.10: Forest Fires: AUC − PRIV estimates

Forest Fires Performance Estimates Ranking ($\mu \pm \sigma$)				
AUC − PRIV	AUC − PRIV$_{rec \geq 0.1}$	AUC − PRIV$_{rec \geq 0.5}$	MAP11	BFM
C	C	B	C	C
0.47409 ± 0.08823	0.38142 ± 0.07782	0.11347 ± 0.02556	0.43098 ± 0.08431	0.49977 ± 0.07749
B	B	C	B	B
0.4267 ± 0.07693	0.34287 ± 0.07603	0.10812 ± 0.03813	0.38585 ± 0.0793	0.49833 ± 0.06807
A ••	A ••	A ••	A ••	A ••
0.36702 ± 0.10149	0.28198 ± 0.0799	0.07284 ± 0.0276	0.3267 ± 0.09042	0.42166 ± 0.07156

Through 1x10 cross-validation, the *paired t-test* asserts that the difference to the best ranked modelling technique is:
• significant at 95% confidence level;
•• significant at 99.9% confidence level.

Model C *is the best ranker according to all the estimators. The exception is when the minimum recall established is 50%. In that case, model* B *becomes better. Still, the differences between* B *and* C *are not judged as significant in none of the estimators. Model* A *is significantly the worst model.*

Regarding Table 4.10 there is another aspect worth to notice. The results show how MAP11 *and* BFM *approximate the performance of the model's PRIV curve provided by* AUC − PRIV. MAP11 *by averaging the precision at 11 pre-specified levels of recall, underestimates the performance of all the three models. On the contrary,* BFM *does an optimistic estimate of the models* AUC − PRIV *performance. Moreover, according to this metric,* C *and* B *have similar performance estimates. The reasoning is that the best F-measure is found at low levels of recall where these two models have identical PRIV curves. Thus, if we were interested in the model's performance on the PRIV curve,* MAP11 *is preferable to* BFM.

4.5 Experimental Study

In this section, we present an experimental study conducted for testing the new proposed utility-based metrics on the prediction of rare extreme values. The experiments are divided by decision-making process and ranking analysis. In the first group of experiments, and as we have previously presented in Torgo and Ribeiro (2009), we use the stock market domain to illustrate the effectiveness of the proposed precision and recall statistics in the context of the prediction of rare extreme returns of a set of stocks. In the second group experiments, we present the PRIV curves obtained by different modelling techniques in the context of predicting harmful algae blooms (Ribeiro and Torgo, 2008a) and evaluate its respective AUC − PRIV. The purpose of this study is to illustrate both the visualization advantages brought by our curves as well as the danger of using standard regression evaluation statistics in this type of problems.

4.5.1 Stock Market Forecasting

Forecasting in the context of financial markets is a rather challenging task, which was already addressed by many studies. The goal of forecasting the correct trading action at a specific moment, e.g. at the end of each day, is directly related to the expected variation of prices in the following days. If prices are expected to raise the decision to buy or uphold is likely to be the correct

call whereas if prices are expected to decrease, the decision to sell is a more sensible one. The combination of price forecasting and trading decision can be turned into a modelling problem, that will support the decision process.

In our experiments we used the standard daily quotes of four companies: International Business Machines (IBM), Coca-Cola (KO), Boeing (BA) and General Motors (GM). This daily data was obtained from Yahoo finance [*2] and it contains the usual daily quotes (Open, High, Low, Close and Adjusted Close) and volume information. Most studies using this type of data focus on predicting the Adjusted [*3] Close prices of the stocks. Namely, the usual procedure consists of taking either the h-days returns of these prices as the basic time series to predict or alternatively the log returns. We used the first of these alternatives in our experiments. The h-days returns of the close prices can be defined as,

$$R_h(t) = \frac{Close(t) - Close(t-h)}{Close(t-h)} \tag{4.14}$$

Using this time series of returns we defined a prediction task that predicts the future value of these returns, $R_h(t+h)$, using a set of p previous values of the time series (usually known as an embed of the time series). In our experiments we used an embed of 24 days back of the $R_h(t)$ variable. This modelling task was selected without any particular concern on whether this was the best setup for predicting future returns. That is not our main goal here. Our objective is to compare alternative modelling techniques on the same stock market prediction problems and check the model rankings we obtain when using both the standard evaluation metrics and our new proposals. Our hypothesis is that the model rankings obtained with our metrics are "better" from the perspective of the application objectives, which are being accurate at the rare extreme returns that are the ones where profitable trading can take place. Using this approach we obtained datasets for the prediction of 1-, 3- and 5-days ahead returns of the four companies used in our study, i.e. 12 regression tasks.

The used quotes data covers the period from 1970-01-02 till 2008-07-11, in a total of 9725 daily sessions.

In order to provide an accurate estimate of the statistics that we will use to compare our alternative models we have divided the period mentioned above in two main consecutive time windows. The first spans from the first date till 1990-01-01. The second time window goes from this latter date till 2008-07-11. The first time window (first 20 years) will be used for obtaining the prediction

[*2] http://finance.yahoo.com
[*3] For stock splits and dividends.

models, while the second window (around 18 and a half years) will be used to evaluate and compare the models.

In what concerns the prediction models, we have used 3 mainly different modeling techniques: multivariate adaptive regression splines (mars), support vector machines (svm) and random forests (randomF). These base techniques were tunned, thus producing several modeling variants as we specify next.

The package `earth` (Milborrow et al., 2010) of R has a re-implementation of MARS Friedman (1991) done by Trevor Hastie and Robert Tibshirani. We have used this system in our experiments. Regarding model tuning we have considered 8 variants formed by different combinations of the parameter setting the penalty for extra degrees of freedom (parameter `degree` which was used with values 1,2), and of the parameter specifying the forward stepwise stopping threshold (parameter `thresh` that was tried with values 0.01, 0.005, 0.001 and 0.0005).

Package `e1071` (Dimitriadou et al., 2010) of R includes a function implementing SVMs. This implementation provides an interface to the award-winning `libsvm` library by Chang and Lin. We have considered 16 variants of SVMs during our model tuning experiments. They include different values for the parameter `cost` (tried values 400, 500, 600 and 700) and `gamma` (tried values 0.01, 0.005, 0.001 and 0.0005). The former is a constraints violation parameter, while the latter is the radial basis function kernel parameter.

Package `randomForest` (Liaw and Wiener, 2002) of R includes a function that implements random forests Breiman (2001) based on original Fortran code by L. Breiman and A. Cutler. We have considered 10 variants of these models by setting the parameter `ntree`, which controls the number of trees in the ensembles, to values from 50 to 500 in steps of 50.

4.5.1.1 The Results

We have obtained the 34 model variants using the experimental methodology described before on the returns data sets. The main hypothesis that we are trying to check is that the model rankings obtained by using our proposed metrics are significantly different from the rankings obtained with standard regression statistics. Moreover, that these rankings obtained with our metrics are clearly advantageous in terms of the application preference bias, which in this case is related to having good "signals" of rare and extreme movements of the markets.

In terms of our evaluation framework we have used the following settings. The target event defined for each stock was based on the notion of rare extreme values given in Section 4.2.2. We have

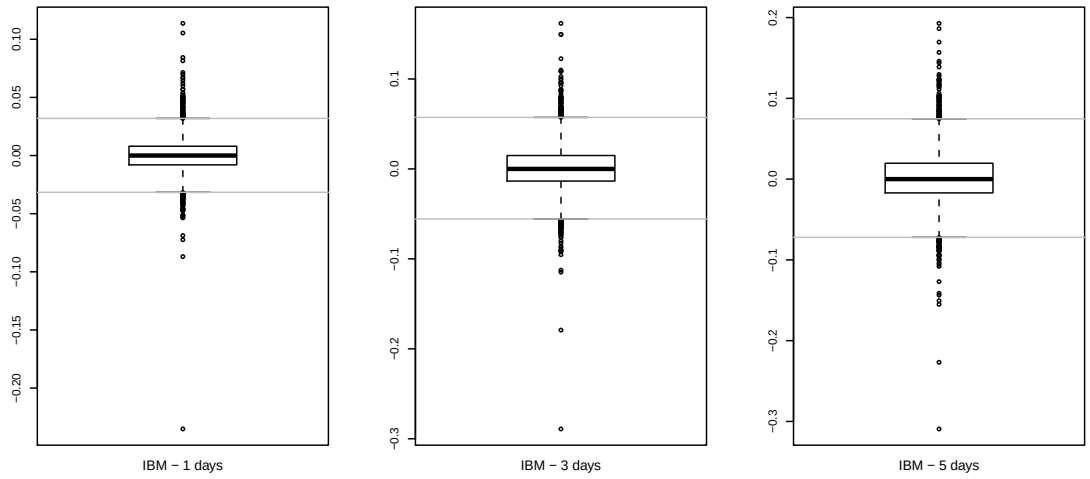


Figure 4.6: Trading thresholds for 1, 3 and 5 days ahead of IBM.

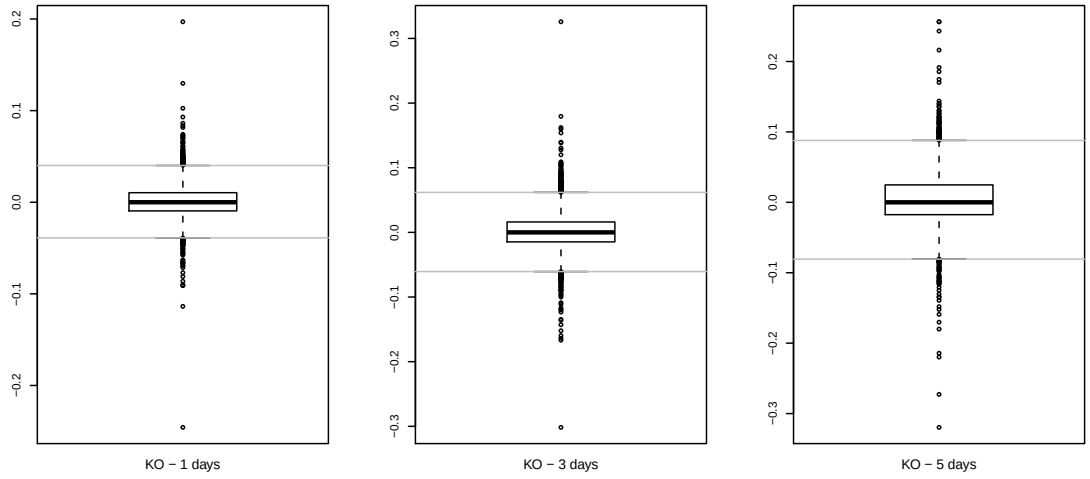


Figure 4.7: Trading thresholds for 1, 3 and 5 days ahead of Coca-Cola (KO).

obtained different thresholds for the notion of high or low extreme returns, which will be the basis for the trading decisions, depending on the stock and on the target variable (the h-days ahead returns), as it is possible to confirm in the Figures 4.6- 4.9 below.

The first results we show are from an experiment designed to test the hypothesis concerning the difference between the rankings obtained with the different metrics. We have used the MAD statistic as a representative of the "standard" approaches, and the composite Fm (with $\beta = 0.5$ that gives twice importance to precision compared to recall, as inaccurate trading signals may be costly) as representing our proposals. For all 12 experimental setups (4 companies and 3 forecasting scenarios), we have obtained the two model rankings according to these two statistics.

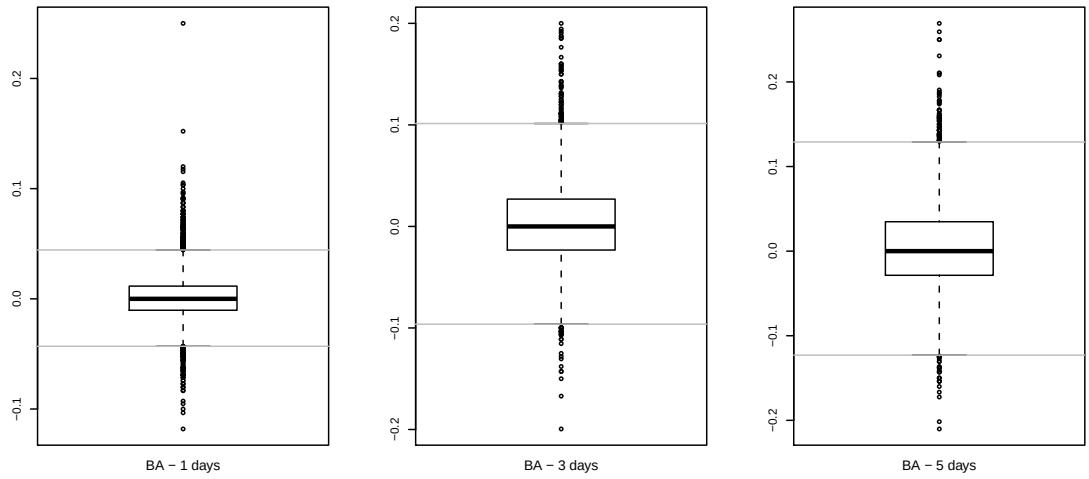


Figure 4.8: Trading thresholds for 1, 3 and 5 days ahead of Boeing (BA).

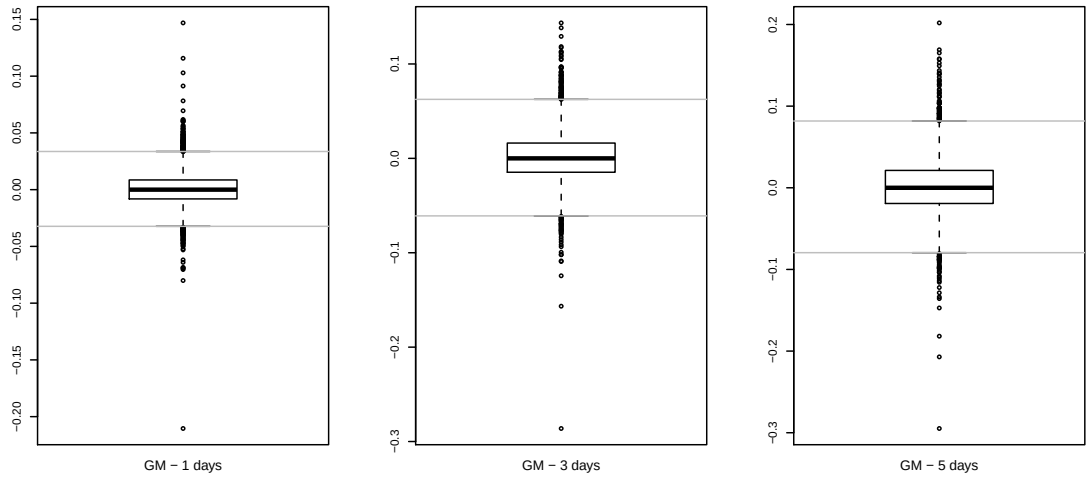


Figure 4.9: Trading thresholds for 1, 3 and 5 days ahead of General Motors (GM).

In Table 4.11, we present the best model according to MAD and Fm. The table includes two lines of results for each task. The first includes the results of the model chosen by MAD, while the second presents the results of the best model according to Fm. In the column "top" of the table we have the ID of these best models - a number between 1 and 34 as this is the number of model variants being compared. Columns # buy and # sell indicate how many times the models issue predictions above (below) the threshold for extreme returns shown in the previous figures. These would correspond to trading signals if we were to use these models for decision making. Columns Ret_{buy} and Ret_{sell} give the average return of "buy" and "sell" signals issue by the models. These provide a quality indicator of the models trading signals. As it is possible to observe, the models

chosen by MAD are conservative, as they do not issue almost no signals. MAD chooses the models that optimize the performance on the most common values of the target variable, and these are the small returns, thus this is not a surprising result. These are not the type of models that interest the end user. On the contrary, the models chosen by Fm take more risks. Moreover, these later models also fulfill the overall application requirements as they generally obtain positive returns for buy signals and negative returns are obtained for sell signals.

Table 4.11: Best models according to MAD and $Fm_{\beta=0.5}$: performance estimates and trading signals.

comp	hday	top	MAD	precision	recall	$Fm_{\beta=0.5}$	# buy	Ret_{buy}	# sell	Ret_{sell}
IBM	1	27	0.013	0.411	0.003	0.015	10	0.022	0	
IBM	1	4	0.024	0.632	0.188	0.429	224	0.023	185	−0.023
IBM	3	27	0.023	0.000	0.000		0		0	
IBM	3	1	0.034	0.553	0.160	0.371	220	0.037	174	−0.038
IBM	5	27	0.030	0.000	0.000		0		0	
IBM	5	3	0.047	0.596	0.148	0.371	177	0.054	165	−0.054
KO	1	28	0.011	0.846	0.001	0.007	1	0.048	0	
KO	1	4	0.016	0.545	0.166	0.374	117	0.026	103	−0.023
KO	3	28	0.019	0.801	0.003	0.013	3	0.048	0	
KO	3	4	0.027	0.638	0.107	0.320	84	0.042	69	−0.039
KO	5	28	0.025	0.547	0.005	0.025	1	0.031	7	−0.051
KO	5	4	0.032	0.642	0.100	0.308	72	0.058	54	−0.055
BA	1	27	0.014	0.000	0.000		0		0	
BA	1	4	0.018	0.708	0.102	0.323	58	0.032	62	−0.029
BA	3	27	0.024	0.000	0.000		0		0	
BA	3	4	0.030	0.696	0.086	0.287	32	0.072	26	−0.063
BA	5	29	0.030	0.581	0.001	0.006	1	0.093	0	
BA	5	2	0.036	0.412	0.067	0.203	44	0.074	37	−0.067
GM	1	27	0.016	0.501	0.003	0.013	9	0.026	0	
GM	1	1	0.027	0.670	0.180	0.434	221	0.027	245	−0.025
GM	3	27	0.028	0.000	0.000		0		0	
GM	3	3	0.046	0.627	0.150	0.383	216	0.047	175	−0.042
GM	5	27	0.036	0.000	0.000		0		0	
GM	5	4	0.059	0.657	0.137	0.373	179	0.062	162	−0.055

Table 4.11 shows that the best models according to each metric are rather different. What about the other positions in the rankings of the 34 model variants that were considered for each of the 12 prediction tasks? We have also carried out that analysis and present the results in a graphical form. The complete set of graphs obtained for is available in Appendix A. Generally, there is a similar results trend in all 12 prediction tasks. We have selected one setup that is shown in Figure 4.10. The results of this comparison of the rankings are shown in two graphs. The graph on the left shows the scores of the best five models according to the two statistics. We should remark that for MAD, lower values are better, contrary to what happens with the Fm. On the X-axis we have the identifiers (a number from 1 to 34) of the top 5 models according to each statistic (the 5 on the left according to MAD and the other 5 according to Fm). Ideally, these two sets of numbers should be

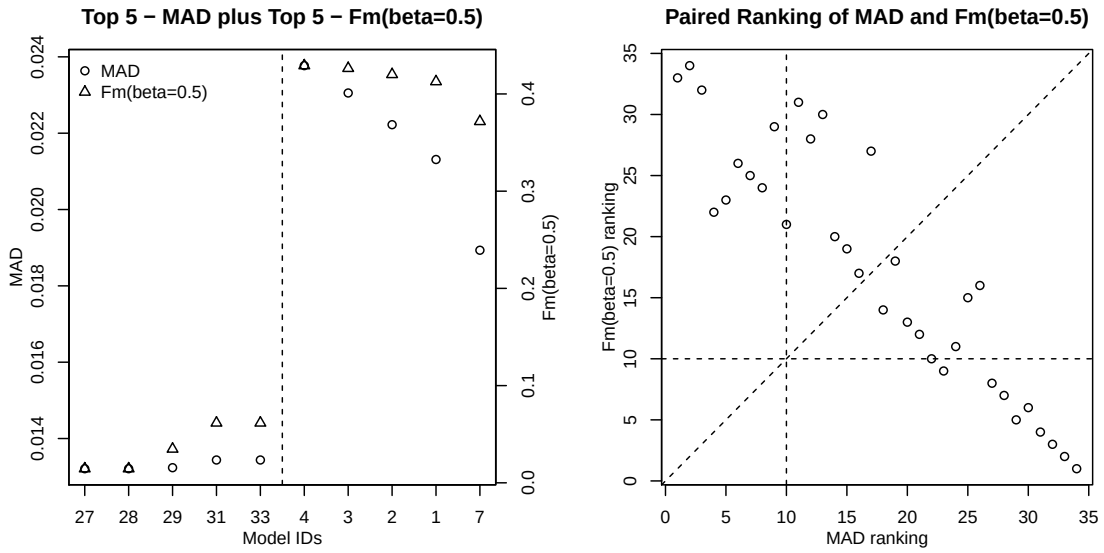


Figure 4.10: The results for 1-days returns of International Business Machines (IBM).

different indicating that the best 5 models according to the two statistics are also different. On the graph we plot the actual values of these 10 models for the two statistics: circles and left Y-scale for MAD; and triangles and right Y-scale for the Fm. The values obtained by some models in some of the metrics are missing from the graph because their scores are out of the scale used in the graph. The second graph on the right shows a global perspective (regards all 34 models) of the two rankings produced by the statistics. On both axis we have the possible ranking positions (from 1 to 34). The coordinates of each of the 34 dots shown on the graphs are obtained using the rank position assigned by MAD (X coordinate), and the corresponding rank position assigned by Fm (Y coordinate). If for any of the 34 models both statistics give the same ranking position, the respective dot should lie in the dashed diagonal line. The vertical and horizontal dashed lines highlight the results for the top 10 rank positions (left of the vertical line, and below the horizontal line) according to the two statistics.

Analysing the results in Figure 4.10, namely the left graph, we observe that the top 5 models according to the two statistics are completely different. Moreover, we see that their concrete scores on the statistics are also very different. For instance the best models according to MAD achieve a much worse score in terms of Fm when compared to the best 5 models according to this later measure. In terms of overall ranking we also observe a general tendency for all ranking positions to be different as most points in the right graph are far from the diagonal. In particular the top 10 models according to MAD are all below the 20th position in the Fm ranking. These results clearly indicate that the two metrics are evaluating very different aspects of the performance of the models.

We have also carried out a *Spearman* statistical test to compare in a more formal fashion the eventual differences between the model rankings. On all 12 data sets we have observed some evidence of disagreement between the rankings, with only 1 lacking proper statistical significance for a 95% significance confidence level (cf. Table 4.12).

Table 4.12: *Spearman* correlation test results of the differences between model rankings.

comp	hday	ρ	p-value
IBM	1	−0.654	0.0000
IBM	3	−0.532	0.0014
IBM	5	−0.629	0.0001
KO	1	−0.587	0.0003
KO	3	−0.528	0.0016
KO	5	−0.662	0.0000
BA	1	−0.425	0.0128
BA	3	−0.178	0.3134
BA	5	−0.385	0.0253
GM	1	−0.672	0.0000
GM	3	−0.530	0.0015
GM	5	−0.704	0.0000

In summary, our experiments have confirmed the hypothesis that the two considered metrics (MAD and Fm based on our proposed recall and precision statistics) often obtain significantly different model rankings on this type of applications. Moreover, we should remark that this experimental setup is not particularly favorable to our proposals. In effect, we are comparing 34 models that optimize some variant of the squared error. This means that these models are not particularly focused on predicting rare extreme values. Even on these conditions we have observed that our proposals are able to detect models that have some ability at predicting rare extreme values. We can expect that the differences would be even more marked if among the 34 models we had some that were particularly competent at predicting rare extremes (e.g. if they were optimizing our Fm instead of squared errors).

These experiments have confirmed the advantages of our metrics. In effect, the models "selected" by MAD almost never issue a single signal during the 18 testing years! On the contrary, the models selected using our metrics issue several trading signals during this period. Still, the accuracy of these signals is far from ideal as expected. This is expectable because: i) the candidate models are optimizing squared errors; ii) the information used to obtain the models (embed of 24 days) is clearly sub-optimal; and iii) predicting stock returns is a very difficult task!

4.5.2 Prediction of Harmful Algae Blooms

Algae blooms are ecological events associated with extremely high abundance values of certain algae. These rare events have a strong impact in rivers ecosystems. By reducing water clarity

and the oxygen levels they degrade water quality and thus a massive death of river fish occurs. This phenomenon has been registered occasionally in several European rivers and, in particular in Douro River, Portugal. For this reason, the early forecast of these blooms is particularly important, especially when potable water is obtained from these rivers.

The data used in our experiments was collected by the public company responsible for potable water collection in the metropolitan Porto area, the second largest city of Portugal. We have used the data resulting from the analysis of water samples collected at Crestuma-Lever dam in river Douro between 1998 and 2003. It includes the identification and quantification of different groups of micro-algae, known as phytoplankton, as well as the analysis of physical-chemical and microbiological parameters.

Identification of the different groups of phytoplankton in water samples requires intensive and expensive manual labour, while the analysis of most physical-chemical and microbiological parameters can be automated by water probes. Our objective is to avoid the expensive manual identification of phytoplankton groups by using models that are able to accurately anticipate their abundance based on the values of other parameters.

After some pre-processing steps (Ribeiro and Torgo, 2007, 2008a), the observations were aggregated with a bi-week frequency for 6 different groups of phytoplankton that we wished to forecast. Namely, Cyanobacteria, Cryptophyta, Chrysophyta, Dinophyta, Chlorophyta and Diatom. To capture the temporal oscillations and the existing mutual influence among the algae abundance values, we also included, for each observation, the values of the previous two weeks of each group of phytoplankton their total abundance value and an ecological diversity index. Algae blooms are characterized by an extremely high total value and a low value of diversity, meaning that a few groups of phytoplankton dominate.

In our application, the target variable is the abundance value of a group of phytoplankton and the predictor variables are a set of physical-chemical and microbiological parameters together with the previous values of the abundance of the groups of phytoplankton, their total value and a ecological diversity index. We have thus 6 different multiple regression problems: one for each group of phytoplankton.

Regarding the experimental methodology, given the time dependency between the cases, we have used a Monte Carlo Simulation to obtain the estimates of our evaluation statistics. For each iteration of the Monte Carlo simulation, a point in time, t, is randomly selected within the available period of data ensuring that there is a pre-specified past window of data to use for training, $t-t_{train}$, and a pre-specified future period of data to use for testing, $t + t_{test}$. Once the model is trained, the predictions for the test period are obtained by a *sliding-window* technique. After each t_{new}

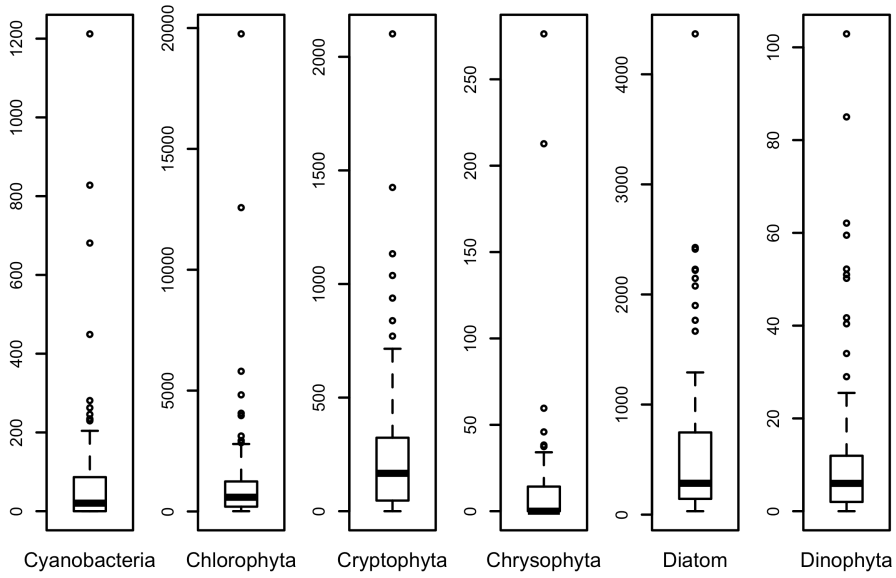


Figure 4.11: Harmful Algae Blooms Distribution Values

predictions the training window is slided and a new model obtained. The sliding process is repeated until all the predictions for the test period are obtained, i.e. we reach t_{test}. In our experiments we have randomly selected the data points from the available 4.5 years of data, and have used two years as training window size and 9 months for testing window. Regards the process of sliding window a new model is re-learned every month.

Regarding prediction models, we have used 4 different approaches: regression trees (cart), random forests (randomF), support vector machines (svm) and multivariate adaptive regression splines (mars). For each base technique we have used several parameter variants to ensure a fair comparison.

Once again we have used the implementations available within the R software. The packages used for random forests, support vector machines and multivariate adaptive regression splines were the same indicated in the previous section. For the regression tree we have used the `DMwR` package (Torgo, 2010) of R, which uses an implementation of the system CART (Breiman et al., 1984). We have considered two alternatives to select the best-sized tree using the standard error parameter `se` parameter equal to 0 and to 1. For random forest, we have considered 2 variants with respect to the number of trees in the ensemble: one with 50 trees and other with 500 trees. For support vector machines, we have considered the values 100 and 500 for the parameter `cost` and the values 0.01 and 0.0005 for the radial basis function kernel parameter `gamma`. Finally, for the multivariate regression splines we have used the parameter setting the penalty for extra degrees

of freedom `dg` with values 1,2, and for the parameter specifying the forward stepwise stopping threshold `thr` the values 0.01, 0.0005.

Contrary to what was done at the Stock Market Forecasting example, the goal here is to get not a fixed-point evaluation, but a ranking analysis of the performance of the models.

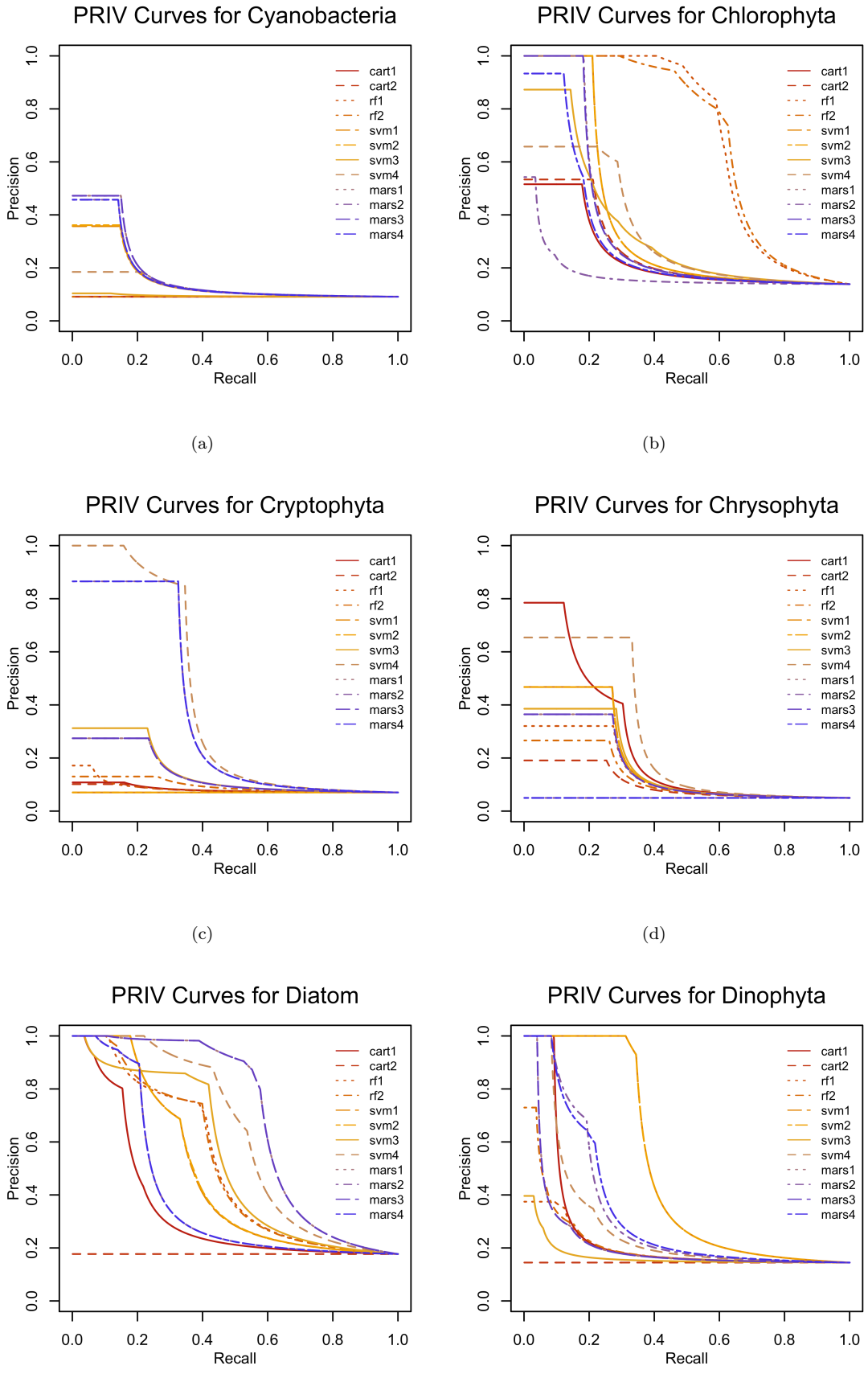
In Figure 4.12, we present the set of pooled PRIV curves obtained by the 8 model variants, one for each of the 6 micro-algae.

Overall, we can conclude that the models do not perform very well for high levels of recall. Moreover, when recall is nearly zero, the variance of the precision among the models can be very high. But these are expectable behaviours in an imbalanced domain such as the prediction of a rare target event. PRIV curves can actually express the dominance of a model for low levels of recall. There are models whose precision is limited by their recall ability, while others show an increased precision for low as well as for high values of recall. In fact, this is one of the main features of these curves that are not possible to visualize through ROC curves. From an overall analysis of these PRIV curves, and considering the set of selected models, we can rule out some of them from our ranking analysis. This is the case of model cart2 for almost all the micro-algae, randomF1 and randomF2 for `Cyanobacteria`, mars2 for `Chlorophyta`, mars4 for `Chrysophyta`, etc. On the contrary, we should make sure to include models like svm1 and svm2 for `Chlorophyta`, svm4 and svm s for `Dinophyta`, etc. For a more precise comparison of the different models we have to calculate their AUC − PRIV and ran a significance t-test for comparison against the top ranked model. Table 4.13, shows the top 3 models obtained for each of the micro-algae.

As it was already observable from the PRIV curves, there is no single model that can be considered as the best. In fact, all the tried models appear at the top 3 ranked models, with the exception of cart2, the CART regression tree with no pruning. Still, from a global perspective, we can say that svm achieve the best results. There are a few exceptions, however. This is the case of random forests for `Chlorophyta` and for `Diatom` above a 0.5 recall level. In effect, the performance of the models for the `Diatom` alga is rather different from the others. For instance, mars, the modeling technique with the worst PRIV curve for some micro-algae, has the best PRIV curve for this micro-alga. Finally, looking at all the PR curves we observe a few critical recall points, which seem to be crucial to the models performance, and thus deserve a further inspection.

4.6 Conclusions

In this chapter we have presented a study on the prediction of rare values of a continuous target variable. In particular, we have addressed the prediction of rare and extreme values that are often



(a) (b)

(c) (d)

(e) (f)

Figure 4.12: PRIV curves obtained for the prediction of harmful algae blooms.

Table 4.13: Harmful Algae Blooms Prediction: AUC − PRIV estimates

	Top 3 Performance Estimates Ranking ($\mu \pm \sigma$)			
	AUC − PRIV	AUC − PRIV$_{rec \geq 0.5}$	MAP11	BFM
Cyanobacteria	svm1	svm1	svm1	svm4
	0.11591 ± 0.1739	0.04511 ± 0.06699	0.11062 ± 0.16534	0.13898 ± 0.20084
	svm2	svm2	svm2	randomF2
	0.11564 ± 0.17351	0.04507 ± 0.06694	0.1104 ± 0.16501	0.13806 ± 0.19979
	mars3	mars3	mars3	svm1
	0.11336 ± 0.16833	0.04497 ± 0.06665	0.10911 ± 0.16154	0.13806 ± 0.19979
Chlorophyta	randomF2	randomF2	randomF2	randomF1
	0.44935 ± 0.40834	0.17003 ± 0.17225	0.42703 ± 0.38931	0.44476 ± 0.40698
	randomF1	randomF1	randomF1	randomF2
	0.44701 ± 0.40796	0.16909 ± 0.17423	0.42427 ± 0.3877	0.44318 ± 0.40693
	svm3 •	svm3 •	svm3 •	svm3 •
	0.32343 ± 0.30765	0.09923 ± 0.09267	0.3052 ± 0.29009	0.30869 ± 0.29059
Cryptophyta	svm4	svm4	svm4	svm4
	0.25528 ± 0.31877	0.07152 ± 0.10411	0.23865 ± 0.29675	0.27171 ± 0.32722
	mars2	mars2	mars2	mars2
	0.2182 ± 0.26988	0.04358 ± 0.05024	0.20229 ± 0.24933	0.22808 ± 0.27466
	mars4	mars4	mars4	mars4
	0.2182 ± 0.26988	0.04358 ± 0.05024	0.20229 ± 0.24933	0.22808 ± 0.27466
Chrysophyta	svm4	svm4	svm4	svm4
	0.2122 ± 0.28029	0.08689 ± 0.10861	0.2003 ± 0.26022	0.28107 ± 0.33478
	cart1	cart1	cart1	cart1
	0.18302 ± 0.25799	0.07268 ± 0.10405	0.1738 ± 0.24058	0.25436 ± 0.30463
	svm3	svm3	svm3	svm3
	0.17107 ± 0.21416	0.06042 ± 0.06623	0.16243 ± 0.19959	0.24875 ± 0.28351
Diatom	mars1	mars1	mars1	mars1
	0.62063 ± 0.35065	0.24473 ± 0.15434	0.59351 ± 0.33718	0.60206 ± 0.3614
	mars3	mars3	mars3	mars3
	0.62063 ± 0.35065	0.24473 ± 0.15434	0.59351 ± 0.33718	0.60206 ± 0.3614
	svm4	svm4 •	svm4	svm4
	0.5295 ± 0.29338	0.15933 ± 0.10439	0.49858 ± 0.27848	0.51108 ± 0.27857
Dinophyta	svm1	svm1	svm1	svm1
	0.36992 ± 0.33703	0.13089 ± 0.11141	0.35109 ± 0.31509	0.4067 ± 0.30057
	svm2	svm2	svm2	svm2
	0.36992 ± 0.33703	0.13089 ± 0.11141	0.35109 ± 0.31509	0.4067 ± 0.30057
	svm4	mars4	svm4	cart1
	0.24663 ± 0.20542	0.08485 ± 0.06179	0.23301 ± 0.18787	0.32002 ± 0.22012

Through a Monte Carlo simulation, the *paired t-test* asserts that the difference to the best ranked modelling technique is:
• significant at 95% confidence level;
•• significant at 99.9% confidence level.

regarded as outliers from a statistical perspective, though they are of key importance to several real world applications. Our study was focused on the development of proper evaluation metrics for these tasks, which is a key step in addressing these problems that are also defined as event-based decision problems. Our goal was to show that utility-based regression can act as good support for the decision-making process.

The notions of precision and recall are ideal for addressing these target problems as they focus the evaluation solely on the important events (the rare extreme values). The derivation of Precision-Recall (PR) Curves is also an important aspect as in many applications, more that being particularly accurate, it is important to be able to provide a good rank of the test cases with respect to the target events.

In this context, we have proposed the incorporation of the concept of utility into two types of evaluation measures: i) fixed-point decision measures, where a decision on a target value being an event or not is determined by a fixed-value (e.g. precision and recall); ii) ranking decision measures, where the decision lies on the ranking of the target values relative to the target event (e.g. $AUC - PR$, MAP11).

We have illustrated the use of these metrics in the context of stock market and harmful algae blooms forecasting applications. Namely, we have used our metrics to compare a large set of models in several experimental setups. Our experiments have confirmed that our evaluation metrics provide a significantly different perspective of the performance of the models, when compared to standard evaluation statistics. Moreover, this perspective is more adjusted to the preference biases of this type of applications. Our experimental results have also shown the danger of using standard evaluation metrics in this class of problems.

- 5 -

ubaRules
Utility-based Regression Rule Ensembles

"All human knowledge takes the form of interpretation."
Walter Benjamin (1892 - 1940)

Cost-sensitive learning is a key technique for addressing many real world data mining applications. We have already seen that this technique is also useful in several regression domains with non-uniform costs and benefits across the domain of the continuous target variable. Given the nature of this type of applications, the relevance of the values within specific ranges of the continuous target variable often imply costly and difficult decisions. This means that, apart from obtaining accurate predictions, it is of key importance to domain experts to have an interpretable model. In this chapter we introduce ubaRules, a rule-based regression system biased by an utility-based metric. We show some of the output rules of our system for applications where the goal is to get accurate and interpretable predictions.

5.1 Main Objectives

In Chapters 3 and 4, we have defined a proper evaluation framework based on the concept of utility. In this chapter we describe a learning system that obtains models optimizing the criteria we have proposed in these chapters. ubaRules is an utility-based regression rules learning system that produces models for our target applications.

Recapitulating, our main challenges for these applications are:

- to express the preference bias of the users of these applications;
- to evaluate/compare/select models in the application context;

- to obtain models that are biased towards these objectives;
- and to explain the models' predictions.

So far, we have presented how the application goals can be expressed through an utility-based metric so that a proper evaluation, comparison and selection of alternative models can be performed.

Throughout this chapter we will address the last two challenges: obtain models biased according to the specified utility-based metric and, at the same time, obtain predictions that are interpretable to the domain user. Interpretability is of key importance to our target applications as relevant values are usually associated with costly/beneficial decisions. In this context, we have decided to select a rule-based formalism to represent our models. Rules are considered to have strong explanatory power mainly due to their modular characteristics. In this context, we describe ubaRules: a regression rule learning system that is able to provide accurate predictions at the more relevant values of the target variable.

Our system has its foundations on techniques of rule learning and ensemble schemes. In Section 5.2, we briefly overview some work related to these techniques. Our proposal is described in detail in Sections 5.3 and 5.4. As the system still has a lot of space for improvements, we propose some upgrades in Section 5.5 as further work. Finally, in Section 5.6, we summarize our work on utility-based regression rules.

5.2 Background and Related Work

In this section we establish the relationship between our algorithm and the existing research on both regression rule learning and ensemble schemes. Our goal is to learn an ensemble of regression rules for a given regression task with possibly non-uniform relevance for the target variable. The model should be able to give both accurate and interpretable predictions for the target variable, particularly at its highest relevance regions.

Whenever the interpretability of a model is regarded as an important aspect to consider, rules are one of the first approaches that are considered. In effect, due to their simple `if-then` form, rules are one of the most understandable way of expressing a prediction model. For this reason, rule learning has always been a key research area within machine learning. Most of the rule learning methods (Flach and Lavrac, 2003) follow a *sequential covering strategy* approach. According to this strategy, the rules are induced directly from the examples - one rule at a time and each one of

them covering a part of the examples. The examples are eliminated as they get covered, and the procedure is repeated until no example in the training set is left. The result is an ordered rule set where rules are weighted by their error estimates. Most research on rule learning systems focus on classification tasks (e.g. CN2 (Clark and Niblett, 1989), RIPPER (Cohen, 1995)). Regarding regression, there is still a small number of rule induction systems. Weiss and Indurkhya (1995) have developed a regression rules system (SWAP1R) that learns rules by means of classification. The target values are first clustered into a set of bins, which are then used as classes. With a similar purpose Torgo and Gama (1997) proposed the RECLA system. Through this system, a regression task is first converted into a classification problem and then addressed by some classification system. In particular, regression rules can be learned if we use a classification rule learning system as base learner. Nevertheless, and as we have argued before, addressing a regression problem through a classification approach results, inevitably, in some information loss. There are other systems that induce regression rules directly from the examples in an iterative covering approach, such as FORS (Karalič and Bratko, 1997), R^2 (Torgo, 1995), Cubist [*1] and M5rules (Holmes et al., 1999). The FORS system is alike the regression version of CN2 system but using a first order logic formalism. The R^2 system has the particularity of building regression models in the consequent part of the rules. Based on M5 system (Quinlan, 1992, 1993), which produces regression model trees, Quinlan proposed Cubist. This system generates an unordered set of regression rules with linear models in the consequent part. Similarly to the M5 system, Cubist can also use the k-nearest neighbour predictions to obtain its final predictions. M5Rules (Holmes et al., 1999) builds a decision list for regression problems also using the model trees created by M5 (Quinlan, 1992). Still, the strategy is to build repeatedly a model tree and, in each iteration, make the "best" leaf into a rule.

Although most popular rule learning algorithms are based on the *sequential covering strategy*, another approach for rule induction appeared more recently: *rule ensembles strategy*. This approach is a generalization of the *sequential covering strategy*, because it sequentially adds rules to the ensemble, independently of the rules that already are in it. A typical procedure of algorithms that follow this strategy to build the rule ensembles is to derive the rules from an ensemble of decision trees. From the rule set, the algorithm selects a subset of rules, based on an optimization procedure like the minimization of the loss function, and assigns each rule a suitable weight. The predictions are given by the weighted vote of all the rules that cover each example. Though large rule sets result from the ensemble, which makes the rule selection essential for the interpretability of the models, the resulting models have shown good accuracy results when compared with non-interpretable models. The goal of the rule ensembles algorithms is to achieve a good tradeoff between interpretability and accuracy. As examples of algorithms that learn rule

[*1] Cubist is a commercial application developed by Ross Quinlan. Some information about the system and a free test version are available at `http://www.rulequest.com`

ensembles, we can refer to SLIPPER (Cohen and Singer, 1999), LRI (Indurkhya and Weiss, 2001), ENDER (Dembczynski et al., 2010), R-PREV (Ribeiro and Torgo, 2006) and RuleFit (Friedman and Popescu, 2008), among others. In effect, the origin of this strategy is intimately related to the appearance of successful algorithms like Random Forests (Breiman, 2001), which are based in ensemble schemes like bagging (Breiman, 1996) and boosting (Schapire, 1999).

Ensemble learning is a powerful learning scheme (cf. Breiman (1996, 2001); Friedman and Popescu (2008); Schapire (1999)), namely in cost sensitive domains. This technique combines multiple learners towards a better performance than what could be obtained by any of the single constituent learners. Breiman (1996) claimed that decision trees are good candidates for these constituent/weak learners. If decision trees are known for its interpretability, they are also very unstable. Small changes in the training data can cause big changes in the tree structure, specially if it affects a decision in the top nodes. In this sense, any error committed in upper splits is propagated down and affects all splits below them. Breiman (1996) has shown that by using ensembles of decision trees, we can maintain their advantages and, at the same time, suppress their instability making their accuracy increase dramatically. Based on this results, successful algorithms like Random Forests (Breiman, 2001) were developed. Regardless of the chosen constituent learner for the ensemble, there are two main ways of combining all the produced models in the ensemble: bagging (Breiman, 1996) and boosting (Schapire, 1999). They both share the same objective: run several iterations of the same learning algorithm on different distributions of training examples, so that smaller prediction errors are obtained by using multiple learners. There is one fundamental difference between them, which regards the training set used on each iteration. Bagging obtains each training set based on an independent sampling with replacement, whereas boosting obtains each training set based on the performance of the learning algorithm on the previous iteration. Some studies (e.g. Fan et al., 1999) have confirmed boosting to be an effective technique in non-uniform costs scenarios, such as the detection of outliers or "hard-to-predict" observations. Furthermore, Drucker (1997) has shown that boosting is an effective approach to reduce the prediction error when building a committee of regression trees.

In what concerns our target applications, the "hard-to-predict" observations correspond to high relevance observations. In effect, according to our utility definition (cf. Chapter 3, Section 3.3.2 - page 85) the highest values of both benefits and costs are achieved at the highly relevant values of the target variable. Thus, for these observations, higher errors will be mapped into major prediction costs. Through boosting, we can partially fulfill our goals, as the model performance is iteratively improved over a particular set of hard observations, i.e. the ones that lead to the highest prediction costs. Still, bagging offers a better computational efficiency when compared to boosting, which can be critical for large data sets. Hence, both bagging and boosting techniques are included in our system.

5.3 Generating Ensembles of Utility-based Regression Rules

Given a regression problem, with a specified (possibly non-uniform) relevance of the target variable, our purpose is to produce a model that consists of an ensemble of rules and that fulfills the target problem goals expressed by the utility function.

More formally, let $\mathcal{DS} = \{\langle \mathrm{x}_i, y_i \rangle\}_{i=1}^{N}$be a set of observations, where x_i is a feature vector from the feature space $\mathcal{X}$ composed by the ρ predictor (independent) variables and y_i is an instance of the target (dependent) variable Y with domain $\mathcal{Y} \subseteq \mathbb{R}$. The model $f : \mathcal{X} \rightarrow \mathcal{Y}$, which describes the concept underlying the set of observations, is a linear combination of base learners that constitute the ensemble. Hence, the model f is designated as *generalized additive model* (Hastie and Tibshirani, 1990) and defined as:

$$f(\mathrm{x}) = a_0 + \sum_{m=1}^{M} a_m \, f_m(\mathrm{x}) \tag{5.1}$$

where M is the number of the iterations in the ensemble, $f_m(\mathrm{x})$ is the base learner on a sample of input variables x of the training data, and $\{a_m\}_{m=0}^{M}$ are the parameters that specify the linear combination of the base learners.

Our goal is to learn the best approximation to f, so that the obtained approximation, $\hat{f}$, is a model that maximizes our utility function as preference criterion. In this context, and assuming that U_{ϕ}^{p} is the utility function (cf. Definition 3.20 - page 92), the objective is to find a regression model $\hat{f}$, with respect to a set of instances $\langle \mathrm{x}, y \rangle \in \mathcal{X} \times \mathcal{Y}$ drawn from a distribution D, that maximizes the expected utility, defined in Definition 3.22 (page 95) as

$$E_U(\hat{f}_{\Omega}, D) \;:=\; E_{\langle \mathrm{x},y \rangle \sim D} \left[\, U_{\phi}^{p}(\hat{f}_{\Omega}(\mathrm{x}), y) \, \right] \tag{5.2}$$

Based on the expected utility, we establish a utility-based performance measure (e.g. MU, NMU, Fm, AUC-PRIV), here generally designated by $\mathcal{U}$, as the preference criterion for our model.

Our proposed system ubaRules for learning utility-based regression rules ensembles is outlined in Algorithm 5.1. It consists in two main steps: generation of different regression trees, which are converted to rule ensembles, and selection of the best rules to include in the final ensemble. The utility function is used as criterion at several stages of the algorithm. Namely, at model selection, ensemble scheme and rule selection, as we will explain next.

Algorithm 5.1 ubaRules($\langle \mathcal{DS}, \phi \rangle, \mathcal{U}, \Lambda$): the main algorithm.

Input: a regression problem described by a data set $\mathcal{DS}$ and a relevance function ϕ, an utility metric $\mathcal{U}$ and a set of ensemble parameters Λ.

Output: Utility-based Regression Rules Model $f^*(\mathrm{x})$.

1: $f_M(\mathrm{x}) \leftarrow$ ubaRulesEns($\langle \mathcal{DS}, \phi \rangle, \mathcal{U}, \Lambda$) // Algorithm 5.3 or Algorithm 5.4
2: $f^*(\mathrm{x}) \leftarrow$ ubaRulesSel($\langle \mathcal{DS}, \phi \rangle, f_M, \mathcal{U}, \Lambda$) // Algorithm 5.5
3: **return** $f^*(\mathrm{x})$

5.3.1 Regression Trees as Base Learner

Similarly to RuleFit (Friedman and Popescu, 2008) or Random Forests (Breiman, 2001), ubaRules starts by generating an ensemble of decision trees.

A decision tree is a structure that connects a set of nodes defined by recursively partitioning the data by means of splitting tests over the predictor variables. The terminal nodes, also called leaves, represent the partitions of data formed by the tree, and contain the predictions of the tree for the target variable.

There are three crucial aspects that guide the growth of a decision tree:

(i) the split criterion to find the best splitting test;

(ii) the stop criterion to establish when a node should not be split any further;

(iii) the function to assign the prediction value to each terminal node.

Starting with a single node at the top, also called root, the tree grows by recursively partition the data according to its split criterion (i) until a stop criterion (ii) is reached. At that stage, a leaf is created and a prediction model (iii) is obtained with the respective data partition. The paths from the root node to each of the leaves form the set of rules that will be used by the decision tree in its predictions.

In our algorithm, we used CART (Breiman et al., 1984) system [*2], which implements binary splitting tests on a single predictor variable. This means that a split test for a node t with a partition of data D_t results in only two nodes: a left subnode t_L, which contains the partition of cases D_{t_L} that satisfy the binary test; and a right subnode t_R, which contains all the remaining cases, i.e. $D_{t_R} = D_t - D_{t_L}$. CART uses the Least Squares (LS) error criterion. According to this criterion, the constant that minimizes the expected value of the squared error is the mean value of the target variable.

[*2] For our system we have used the `rpart` package (Therneau et al., 2008), an implementation very similar to CART that is available for R (R Development Core Team, 2010) software.

In this context, the estimated error for a node t is defined as

$$Err(t) = \frac{1}{n_t} \sum_{D_t} (y_t - \bar{y}(D_t))^2 \tag{5.3}$$

where $\bar{y}(D_t)$ is the mean of the target variable values on the partition D_t of node t, and n_t is the size of that partition.

The error of a split s over the node t is estimated as the weighted average of the errors of its left subnode t_L and right subnode t_R as follows,

$$Err(s,t) = \frac{n_{t_L}}{n_t} Err(t_L) + \frac{n_{t_R}}{n_t} Err(t_R) \tag{5.4}$$

where n_t is the size of the partition of node t, and n_{t_L} (n_{t_R}) is the size of the left (right) partition resulting from the binary split s over the node t.

In order to find the best split, all the predictors are exhaustively searched. From the set of candidate splits S at a node t, the best split s^* is the one that maximizes the decrease in error at t, i.e.

$$s^* = \underset{s \in S}{\operatorname{argmax}} \left(Err(t) - Err(s,t) \right) \tag{5.5}$$

All the error estimates that guide the split criterion, and thus the tree growth, are based on the training sample. These error estimates, so-called *resubstitution estimates*, become more unreliable, as the initial data set is recursively partitioned by the splits into smaller data sets. If no stop criterion is established, a tree can grow until it reaches one observation per leaf. All this process leads the tree to overfit the training data. To minimize the overfitting effect, two techniques are applied in the context of trees: pre-pruning and post-pruning.

Pre-pruning establishes a condition, a stop criterion, that determines when a node should not be split any further. Such condition can be a maximum depth for the tree, a minimum size for the partition of leafs or for the partition of the internal nodes, or even by establishing minimum error threshold, among other requisites. Still, according to Breiman et al. (1984), this procedure does not guarantee us the optimal tree of a given size, with respect to its estimated error. Post-pruning is a more complex pruning technique but also a more reliable one. This procedure grows an initial maximal tree and then prunes the overfitted tree to a "right-sized" tree. But, as we will explain further on, more than avoiding overfitting, this technique provides us a set of interesting subtrees.

5.3.2 Best-Sized Regression Tree

Regression trees are efficient and interpretable but as they grow in size their interpretability diminishes. This means that, as trees grow, some of the model interpretability is traded off for better performance.

The length of the path from the root until each of the leafs of the tree, determines the maximum number of logical conditions appearing in the rules derived from that tree. In this sense, larger trees typically derive more complex rules, in terms of the number of factors (variables) that compose them. For this reason, and to maintain the interpretability of our models, our trees should be compact and still accurate. If we control the size of the trees, we control the complexity of the rules.

The *error complexity pruning* is the post-pruning method used in CART (Breiman et al., 1984) to avoid *overfitting* of the trees. CART computes a large tree T_0 to fit the training data, by allowing the splitting process to continue until all terminal nodes reach some stopping criterion. Next, it obtains from T_0 a nested sequence of subtrees T_i of decreasing size, i.e. $T_0 \ll \ldots \ll T_v$ where T_v is the smallest subtree, which consists of the root node only. This sequence of subtrees is associated with the notion of error complexity of a tree T, which is given by,

$$EC_\alpha(T) = Err(T) + \alpha \cdot |\tilde{T}| \tag{5.6}$$

where $Err(T)$ is the *resubstitution* error of T, $|\tilde{T}|$ is the number of leaves, and α (≥ 0) is the complexity parameter that represents the cost of increasing the size of T.

The value of α controls the degree of pruning with respect to the initial maximal tree T_0. If $\alpha = 0$, then the penalty for having a large number of terminal nodes is null. As α increases, so it increases the penalty per terminal node.

Making each node in T_0 a prune possibility, CART iteratively prunes the initial tree and obtains a sequence of subtrees T_i as follows. Starting with $\alpha = 0$ and $T = T_0$, α is increased so that it would cause a node to be pruned out from T. It then prunes the tree and updates α. The process is repeated until it reaches the root node. This yields to the sequence of subtrees T_i where each tree is characterized by a different complexity value α. To get reliable estimates of each pruned tree, CART evaluates them by cross validation [⋆3].

The goal is to select the "right-sized" tree from all the subtrees, i.e. the tree with a good trade-off between its size and error estimate. In order to do so, Breiman et al. (1984) proposed two processes

[⋆3] For further details on error complexity pruning algorithm see Breiman et al. (1984); Torgo (1999).

of selection: 0-SE rule and 1-SE rule. The first rule selects the tree in the sequence with the lowest estimate of error e^*. The second rule selects the smallest tree in the sequence, whose error estimate is within the interval $e^* + SE(e^*)$, where $SE(e^*)$ is the standard error of the lowest estimate e^*.

In ubaRules we use the CART post-pruning resource to select, from the generated sequence of subtrees, the tree that maximizes an utility estimate. This means that although the trees are grown using a standard error metric, the post-pruning tree selection phase is guided by an utility-based metric that is in accordance with the applications preference biases. We obtain utility-based estimates of the generated subtrees using a similar cross-validation experimental process as in CART. To obtain a more compact and still "good" regression tree, we choose the tree with the maximum utility estimate. As too large trees may overfit the data, we also let the maximum depth d of trees to be a tunable parameter. This procedure is presented in Algorithm 5.2 and illustrated in Example 5.1. The selected regression tree T_{best} is converted into its correspondent rule set, which is formed by one rule per leaf, i.e. $|\tilde{T}_{best}|$ rules.

Algorithm 5.2 ubaTreeRules($\langle \mathcal{DS}, \phi \rangle, w, d, \mathcal{U}$): utility-based CART rules.

Input: a regression problem described by a data set $\mathcal{DS}$ and a relevance function ϕ, a weight vector w, maximum depth d, an utility metric $\mathcal{U}$.

Output: $f_T(\mathrm{x})$ - set of rules extracted from the best post-pruned regression tree.

1: $T(\mathrm{x}) \leftarrow \mathsf{cart}(\mathcal{DS}, w, d, \alpha = 0)$ // `rpart` package (Therneau et al., 2008)
2: $(\mathcal{T}, \mathcal{V}) \leftarrow \mathsf{fetchSubtrees}(T(\mathrm{x}))$ // $\mathcal{T}$ set of all subtrees; $\mathcal{V}$ cross-validation space
3: $T_{best}(\mathrm{x}) \leftarrow \mathrm{argmax}_{T_i \in \mathcal{T}} \mathcal{U}(T_i(\mathrm{x}), \langle \mathcal{V}, \phi \rangle)$ // select best subtree
4: $f_T(\mathrm{x}) \leftarrow \sum_{k=1}^{|\tilde{T}_{best}|} r_k(\mathrm{x})$ // convert to rules
5: **return** $f_T(\mathrm{x})$

Example 5.1. Selection of the best utility post-pruned tree: prediction of NO_2 emissions.

In this example, we return to the outdoor air pollution prediction task (Aldrin and Haff, 2005) presented in Chapter 3, Section 3.2.1 (page 67). The goal of this application is to forecast high and potentially nocive concentration of NO_2 *emissions registered hourly. The target variable (*`LNO2`*) is the log-transformed concentration of* NO_2 *emissions measured in a traffic road in Oslo. The predictor variables include the logarithm of the number of cars per hour (*`LCarsH`*); the temperature registered 2 meters above ground (*`Temp`*); the wind speed (*`WSpeed`*), the temperature difference between 25 and 2 meters above the ground (*`TempDiff`*); the wind direction in degrees (*`WDir`*); the hour of day (*`Hour`*); and the day number from the start of the study (*`Day`*).*

With the data set (Aldrin and Haff, 2005) of the prediction task describe above, we ran CART with the 1-SE post pruning rule. The result is the regression tree shown in Figure 5.1.

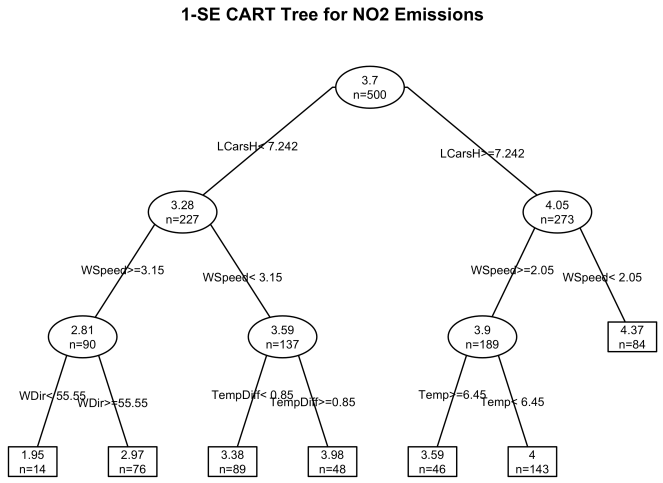


Figure 5.1: CART best post-pruned tree with 1-SE rule.

The specification of 1-SE post-pruning rule indicates that, from all the post-pruned trees and based on the cross-validation error estimates, the final tree should be the smallest tree whose error estimate does not exceeds the minimum error estimate plus 1 times the standard error of the minimum error estimate.

This selection procedure is illustrated in Figure 5.2. In this plot, we have all the subtrees generated by CART establishing $\alpha = 0$ stipulated that the maximum depth is 4. Each subtree is identified by its complexity parameter α on the X-axis. As it is possible to observe, as the complexity parameter decreases towards 0, the size of the tree increases from 1 to 14. For each tree, we have on the Y-axis its cross-validation error estimate and standard deviation. The minimum error estimate is obtained by the tree with size 9. If the selected pruning rule was 0-SE then this would be the returned tree. However, as the 1-SE rule is requested the chosen tree is the smallest tree whose error estimate is below the threshold the minimum error estimate plus its standard estimate. In Figure 5.2 this threshold is signaled by the horizontal line.

Our goal is to change the pruning criterion: instead of pruning by the relative mean squared error as CART, we want to perform pruning by the relative mean utility. In order to do so, we obtain the cross-validated utility estimates of the sequence of post-pruned trees returned by CART. Figure 5.3 shows the mean utility estimates and its standard deviations obtained by cross-validation for the same set of trees of Figure 5.2. Two things are worth to notice over this new set of estimates: (1) the overall objective to the tree selection has changed, we now wish to find the tree that, instead of minimizing the expected error, maximizes the expected utility; (2) as the trees sequence remains the same, i.e. grown by CART following the least squares criterion, it is not surprising that none of them optimizes utility. In effect, in comparison to the cross-validation error estimates, the cross-validation utility estimates indicates a much higher variance.

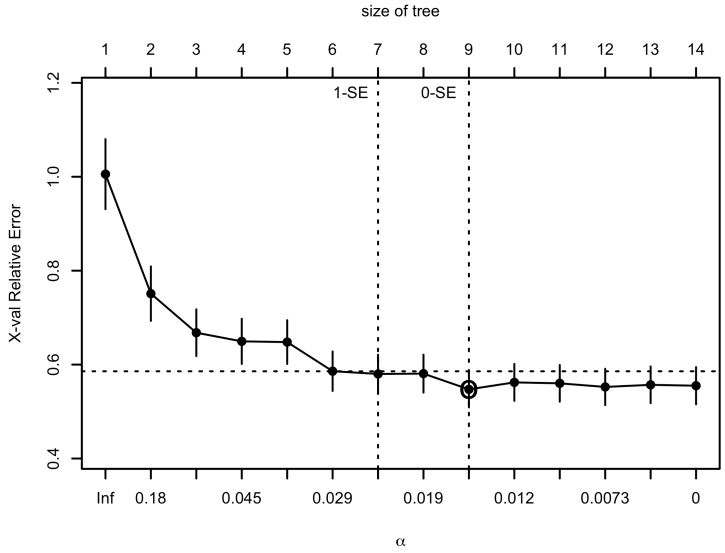


Figure 5.2: Using cost-complexity pruning to select the best-sized tree.

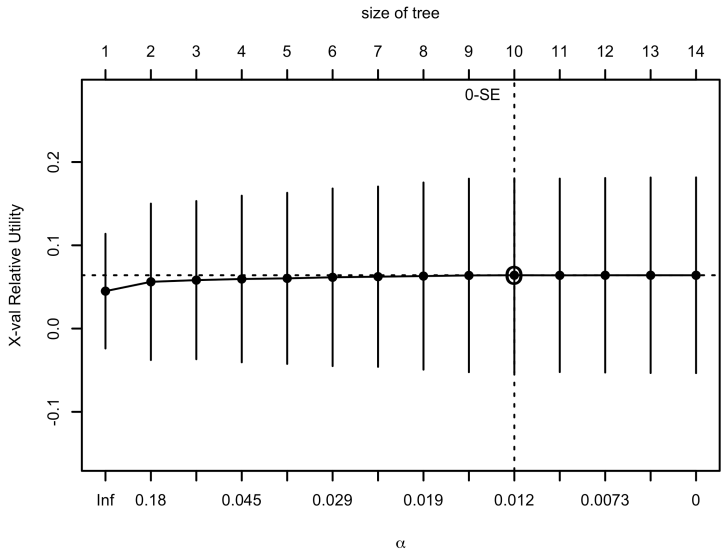


Figure 5.3: Using cost-complexity pruning to select the best-sized tree according to utility.

In this context, if we apply the 0-SE post-pruning rule, which corresponds to the selection of the tree in the sequence with the highest mean utility estimate, we obtain the tree of size 10, shown in Figure 5.4. Thus, for this particular case, the best sized tree would be different whether the objective is to minimize the squared error, as CART does, or to maximize the utility. According to the standard error metric, a tree of size 9 is the better option, whereas according to the utility metric a tree of size 10 is preferable. More specifically, one more split was added to the previous CART tree. This split gives further information over the more relevant values of the target variable, namely the high concentration values of NO_2 *emissions. In this example, it can be noticed that the model that would be chosen by CART is poor when faced with the goals of utility.*

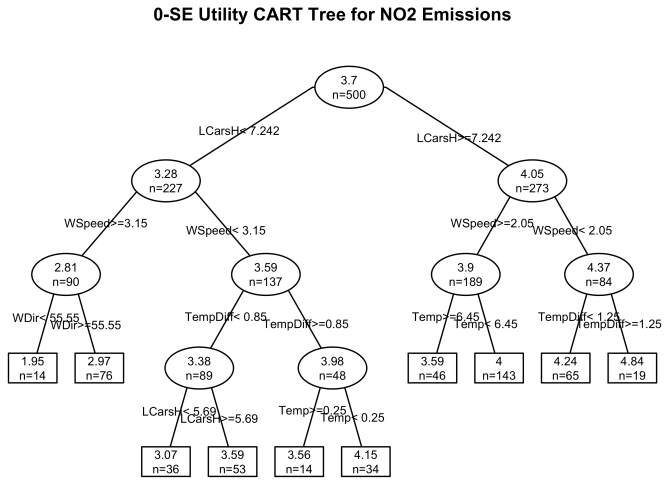


Figure 5.4: CART best post-pruned with the maximum relative utility.

5.3.3 Rule Ensemble Generation

Assuming that the domain of the input variables is $\mathcal{X} := \mathcal{X}_1 \times \ldots \times \mathcal{X}_\rho$, a decision rule $r_k(\mathrm{x})$ is a conjunction of tests of the form $x_j \in \mathcal{X}_j^k$, where x_j is the value of x on input variable j and $\mathcal{X}_j^k$ is a subset of the domain values for input variable j, i.e. $\mathcal{X}_j^k \subseteq \mathcal{X}_j$. Formally, we can define a regression rule $r_k(\mathrm{x})$ as follows,

$$r_k(\mathrm{x}) = \upsilon_k \cdot \prod_{j=1}^{\rho} I(x_j \in \mathcal{X}_j^k) \tag{5.7}$$

where I is the indicator function giving 1 if its argument is true and 0 otherwise, and ρ is the number of input variables and υ_k is the constant prediction of rule k for the target variable Y. If the values of all the input variables of the given example satisfy simultaneously the conjunction of tests specified by the rule, then the rule "fires". In this case, the rule is said to cover the example and returns υ_k as prediction, i.e. $r_k(\mathrm{x}) = \upsilon_k$. If the example does not fulfill the conditions imposed by the rule, then it does not "fire", i.e. $r_k(\mathrm{x}) = 0$. A decision tree T can thus be written as a linear combination of decision rules, as follows,

$$f_T(\mathrm{x}) = \sum_{k=1}^{|T|} r_k(\mathrm{x}) \tag{5.8}$$

where $|T|$ is the number of leafs in the tree.

The rules derived from regression trees are mutually exclusive and exhaustive. This means that given any test case, one and only one rule will be "fired" for its prediction, as it shows Example 5.2.

Example 5.2. Set of rules derived from a regression tree: prediction of NO_2 emissions.

From the regression tree shown in Figure 5.4, 10 rules are derived, one per each leaf in the tree. Each rule is represented by the product of binary decisions over the predictor variables that are associated with all the edges on the path from the root to the leaf. Thus, these rules are formed by mutually exclusive conditions, which means that given any example x_i *only one of these rules will "fire".*

$$
\begin{aligned}
r_1(\mathrm{x}) &= 1.95 \cdot I(\texttt{LCarsH} < 7.242) \cdot I(\texttt{WSpeed} \geq 3.15) \cdot I(\texttt{WDir} < 55.55) \\
r_2(\mathrm{x}) &= 2.97 \cdot I(\texttt{LCarsH} < 7.242) \cdot I(\texttt{WSpeed} \geq 3.15) \cdot I(\texttt{WDir} \geq 55.55) \\
r_3(\mathrm{x}) &= 3.07 \cdot I(\texttt{LCarsH} < 5.69) \cdot I(\texttt{WSpeed} < 3.15) \cdot I(\texttt{TempDiff} < 0.85) \\
r_4(\mathrm{x}) &= 3.59 \cdot I(5.69 \leq \texttt{LCarsH} < 7.422) \cdot I(\texttt{WSpeed} < 3.15) \cdot I(\texttt{TempDiff} < 0.85) \\
r_5(\mathrm{x}) &= 3.56 \cdot I(\texttt{LCarsH} < 7.422) \cdot I(\texttt{WSpeed} < 3.15) \cdot I(\texttt{TempDiff} \geq 0.85) \cdot I(\texttt{Temp} \geq 0.25) \\
r_6(\mathrm{x}) &= 4.15 \cdot I(\texttt{LCarsH} < 7.422) \cdot I(\texttt{WSpeed} < 3.15) \cdot I(\texttt{TempDiff} \geq 0.85) \cdot I(\texttt{Temp} < 0.25) \\
r_7(\mathrm{x}) &= 3.59 \cdot I(\texttt{LCarsH} \geq 7.422) \cdot I(\texttt{WSpeed} \geq 2.05) \cdot I(\texttt{Temp} \geq 6.45) \\
r_8(\mathrm{x}) &= 4 \cdot I(\texttt{LCarsH} \geq 7.422) \cdot I(\texttt{WSpeed} \geq 2.05) \cdot I(\texttt{Temp} < 6.45) \\
r_9(\mathrm{x}) &= 4.24 \cdot I(\texttt{LCarsH} \geq 7.422) \cdot I(\texttt{WSpeed} < 2.05) \cdot I(\texttt{TempDiff} < 1.25) \\
r_{10}(\mathrm{x}) &= 4.84 \cdot I(\texttt{LCarsH} \geq 7.422) \cdot I(\texttt{WSpeed} < 2.05) \cdot I(\texttt{TempDiff} \geq 1.25)
\end{aligned}
$$

Though a tree-based prediction is straightforward, it is very prone to the choices made by the regression tree, which are known to be very sensitive to the given training set (Breiman, 1996). Furthermore, the selection of the "best-sized" tree is highly dependent on the unknown target function (Friedman and Popescu, 2008). Therefore, we have decided to obtain our rule ensemble from a set of utility post-pruned regression trees, as initially outlined in Algorithm 5.1, in the following conditions. Each regression tree is generated from a different training set, obtained by a sampling technique, and with a maximum depth randomly drawn from an exponential distribution as proposed by Friedman and Popescu (2008) for the tree size.

Let $\bar{d}$ be the average maximum depth of all the trees in the ensemble. Then, at each iteration, we generate a maximum depth parameter d by

$$d = 2 + \lfloor \delta \rfloor \tag{5.9}$$

where δ is sampled from a distribution with probability

$$P(\delta) = \frac{1}{\bar{d} - 2} \exp\left(\frac{-\delta}{\bar{d} - 2}\right). \tag{5.10}$$

By proceeding this way, we obtain both small-depth trees, and thus compact rules, and large trees, thus more complex rules. Relative to the established average maximum depth $\bar{d}$, most of trees will have a smaller maximum depth, i.e. $d \leq \bar{d}$, some trees will have a bigger maximum depth, i.e. $d > \bar{d}$, and a few trees will have a much bigger maximum depth, i.e. $d \gg \bar{d}$.

The overall goal is to obtain a more evenly distributed complexity of the rules derived from the ensemble. This will ensure a higher modularity to each rule derived from the regression trees and added to the final rule ensemble model.

At the end of the iterations, for a given test case, more than one rule can then be "fired". In such situations, an average, in case of bagging, or a weighted average, in case of boosting, of the predictions of the fired rules is made as prediction, as we will explain next.

5.3.3.1 Bagging Regression Trees

As we have previously mentioned, Bagging (Breiman, 1996) is an ensemble technique used to improve the accuracy of a single base learner such as regression trees. The strategy is to generate multiple models, aggregating them into a final model by making its predictions the average of predictions from each model. Each model is trained on a bootstrap sample, i.e. a sample chosen uniformly at random with replacement from the original training set and of the same size. Each tree is given a maximum depth that is taken from an exponential distribution, with a minimum value of 2. In ubaRules (cf. Algorithm 5.3), we obtain a set of M stratified bootstrap samples, so that for each sample we can have a similar distribution of values for the target variable and thus a similar set of relevant values. For each sample, we run the regression tree method that selects the best-sized tree according to the utility estimate (cf. Algorithm 5.2) and derive the corresponding set of rules, which are added to the overall rule ensemble.

Our final bagged rule ensemble outputs the average of the predictions made by each of the M models in the committee.

Algorithm 5.3 ubaRulesBagg($\langle \mathcal{DS}, \phi \rangle, \mathcal{U}, \Lambda$): bagging of regression rules.

Input: a regression problem described by a data set $\mathcal{DS} = \{\langle \mathrm{x}_i, y_i \rangle\}_{i=1}^{N}$ and a relevance function ϕ, an utility metric $\mathcal{U}$ and a set of ensemble parameters Λ, where $M \in \Lambda$ is the number of iterations and $\delta \in \Lambda$ is the average maximum depth of the trees in the ensemble.

Output: $f_M(\mathrm{x})$ - Utility-based "Bagged" Rules Ensemble.

1: **for** $m = 1$ **to** M **do**
2: $d \leftarrow$ expDistr(δ) // maximum tree depth drawn from exponential distribution
3: $w \leftarrow$ bootstrap($1, N$) // sample of size N with replacement
4: $f_m(\mathrm{x}) \leftarrow$ ubaTreeRules($\langle \mathcal{DS}, \phi \rangle, w, d, \mathcal{U}$) // Algorithm 5.2
5: **end for**
6: **return** $f_M(\mathrm{x}) \leftarrow \sum_{m=1}^{M} f_m(\mathrm{x})$

5.3.3.2 Boosting Regression Trees

Boosting, similarly to bagging, was originally proposed as an ensemble scheme to enhance the performance of weak learning algorithms. The principle was the same: generate multiple predictions, which were then aggregated by majority/average voting. Still, contrary to bagging, boosting uses the performance estimate of each learner to build a better learner in the following iteration. AdaBoost (Freund and Schapire, 1997) is the most well known boosting algorithm. This method adaptively re-weights the training set based on the error rate of the previous learner. The goal is to progressively enhance the global performance over the training set. Thus, at each iteration, the sampling probability of each case is adjusted so that larger weights are assigned to the cases with bigger prediction errors. By doing so, these will have a higher probability of appearing in the training set of the next iteration.

Even though AdaBoost was first introduced for classification problems by Freund and Schapire (1997), it was then expanded for regression. Shrestha and Solomatine (2004) proposed AdaBoost.RT, a boosting algorithm strictly designed for regression problems. The key issue in such algorithm is the use of a regression threshold (RT) to allow the distinction between a good and a bad prediction, likewise it happens in classification. Hence, according to the authors (Shrestha and Solomatine, 2004), that regression threshold establishes a pre-specified relative error above which any case is considered wrongly predicted. As such, all the cases whose error is below this threshold are considered correctly predicted.

Regarding our target applications, the use of a regression threshold based on a standard error metric related to the absolute deviation, as originally suggested by Shrestha and Solomatine (2004), is not appropriate (cf. Section 3.2.2 of Chapter 3 - page 69). In a non-uniform relevance scenario such as the one that characterizes our target regression problems, standard error metrics do not reflect the performance of a model with respect to the relevance of the values. Nevertheless, the utility of a prediction provides us all the information we need to address the notion of accurate/erroneous

prediction or, correspondingly, the notion of beneficial/costly prediction. More specifically, if the prediction represents a benefit in the context of the application then its utility value is positive. On the contrary, if the prediction represents a cost then its utility value is non-positive.

In this sense, boosting allows our algorithm to focus on the examples that achieve higher costs, which typically are the ones with higher relevance. Our purpose is to boost regression trees in order to create a committee of progressively more powerful regression rules on the prediction of relevant values. Algorithm 5.4 presents the boosting procedure used by ubaRules and that is based on AdaBoost.RT (Shrestha and Solomatine, 2004). Similarly to what we described in the previous section for bagging, in boosting each tree has a maximum depth that is randomly obtained from a exponential distribution, with a minimum value of 2.

Given a M number of iterations, the algorithm starts with an uniform sampling distribution, weighting all the instances of the data set equally with w. From here, it will iteratively update the probabilities of sampling for the next iteration and save the weights that will be assigned to each rule in the final rule ensemble. The goal is to enhance the performance on the harder examples according to the application goals.

At each iteration m, the new predictor $f_m(\mathrm{x})$ created for the data set $\mathcal{DS}$ with sampling distribution w, together with the all the previous $m-1$ predictors and weights, are evaluated. Based on the total weight (ϵ_m) of the instances that obtained costs, it then estimates the confidence it has on $f_m(\mathrm{x})$.

The measure β_m is the inverse measure of confidence in the new predictor. As such, low values of β_m mean high confidence in the predictor. This measure exhibits a quadratic growth with the weight associated to the cost predictions (other growth rates such as linear and cubic may have been chosen - cf. Shrestha and Solomatine (2004)). The sampling probabilities are then updated according to the following. The set of weights associated to beneficial predictions, i.e. $\{w_i^{(m)} : u_i > 0\}$, should be reduced proportionally to the overall cost obtained by the predictor, i.e. β_m. By proceeding this way, there is a smaller probability of instances, that obtained benefits, to be included in the sample of the next iteration. Only the weights of beneficial predictions are updated, thus the weight probabilities for the costly predictions remain the same. Nevertheless, there is a situation we must address in our proposal: the case when all the predictions are considered to bring a benefit. In effect, in the original algorithm AdaBoost.RT (Shrestha and Solomatine, 2004) this situation is reported when no error, above the specified threshold, is committed by the learner. This threshold is typically a small value. Still, obtaining benefits in the context of our target problems may be more easily achieved. For these cases, and in order not to stop the learning iterations too soon, we decided to establish a minimum overall cost of a predictor given by $\beta_m = \min_i \left(w_i^{(m)}\right)^3$. This will make the measure of confidence in the m^{th} predictor to be exceptionally high, when compared to

Algorithm 5.4 ubaRulesBoost($\langle \mathcal{DS}, \phi \rangle, \mathcal{U}, \Lambda$): utility-based boosting of regression rules (based on AdaBoost.RT (Shrestha and Solomatine, 2004)).

Input: a regression problem described by a data set $\mathcal{DS} = \{\langle \mathrm{x}_i, y_i \rangle\}_{i=1}^{N}$ and a relevance function ϕ, an utility metric $\mathcal{U}$ and a set of ensemble parameters Λ, where $M \in \Lambda$ is the number of iterations and $\delta \in \Lambda$ is the average maximum depth of the trees in the ensemble.

Output: $f_M(\mathrm{x})$ - Utility-based "Boosted" Rules Ensemble.

for $i \leftarrow 1$ **to** N **do**
  $w_i^{(1)} \leftarrow 1/N$ // initial weights
end for
for $m = 1$ **to** M **do**
  $d^{(m)} \leftarrow \mathsf{expDistr}(\delta)$ // maximum tree depth drawn from exponential distribution
  $f_m(\mathrm{x}) \leftarrow \mathsf{ubaTreeRules}(\langle \mathcal{DS}, \phi \rangle, w^{(m)}, d^{(m)}, \mathcal{U})$ // Algorithm 5.2
  $f_M(\mathrm{x}) \leftarrow \dfrac{\sum_{l=1}^{m-1} \left(log \frac{1}{\beta_l} \right) f_l(\mathrm{x})}{\sum_{l=1}^{m-1} \left(log \frac{1}{\beta_l} \right)} + f_m(\mathrm{x})$
  for $i \leftarrow 1$ **to** N **do**
    $\mathrm{u}_i \leftarrow U_\phi^p(f_M(\mathrm{x}_i), y_i)$ // calculate the utility of each prediction
  end for
  $\epsilon_m \leftarrow \sum_{i:\mathrm{u}_i \leq 0} w_i^{(m)}$ // sum weights associated to costs made by f_m
  $\beta_m \leftarrow \max \left\{ \min_i \left(w_i^{(m)} \right)^3, \epsilon_m^2 \right\}$ // measure inverse to the confidence in f_m
  // update weights
  for $i \leftarrow 1$ **to** N **do**
    if $\mathrm{u}_i > 0$ **then**
      $w_i^{(m+1)} \leftarrow w_i^{(m)} \beta_m$
    else
      $w_i^{(m+1)} \leftarrow w_i^{(m)}$
    end if
  end for
  for $i \leftarrow 1$ **to** N **do**
    $w_i^{(m+1)} \leftarrow w_i^{(m+1)} / \sum_{i=1}^{N} w_i^{(m+1)}$ // re-normalize distribution
  end for
end for
return $f_M(\mathrm{x}) \leftarrow \dfrac{\sum_{m=1}^{M} \left(log \frac{1}{\beta_m} \right) f_m(\mathrm{x})}{\sum_{m=1}^{M} \left(log \frac{1}{\beta_m} \right)}$

predictors that obtain costs. For the following iteration, sampling probabilities of all the instances are equally reduced. In practice, a sample is obtained with a distribution equal to the previous iteration and a new model is learned.

Our final boosting model outputs the weighted average of the predictions of the models forming the committee, each weight corresponding to the confidence in the model for the prediction of relevant values.

5.4 Selecting the Best Utility-based Rules

The first step of our algorithm is to obtain a set of rules from an ensemble of regression trees. We then try to simplify our model by eliminating some rules from this set using the information regarding their individual utility estimates.

During the transformation of a tree into a set of rules, by performing logic simplifications, redundant conditions in paths are dropped. Therefore, the rule set derived from a tree can have improved readability compared to the paths of the tree. Nevertheless, from an ensemble of regression trees, too many rules are generated, which decreases the model interpretability. From the ensemble of regression trees, we obtain the rule ensemble $\{r_k(\mathrm{x})\}_1^K$. The number of rules in the ensemble is

$$K = \sum_{m=1}^{M} |\tilde{T}_m| \tag{5.11}$$

where $|\tilde{T}_m|$ is the number of terminal nodes for the m^{th} tree.

As the number of rules increases, the 'accuracy' of the model can improve but, at the same time, its interpretability diminishes. Too many rules make the model more complex and its interpretation harder. Therefore, we decided to select the best rules according to the utility estimates. Nonetheless, this rule selection process removes the complete coverage property that results from the mutual exclusivity of a tree, which means that there may exist a test case that is not covered by any of the rules in the final model. For these situations, we have added a default rule $r_0(.)$. This default rule predicts $\overline{Y}$ the mean value of the target variable Y, i.e. the constant that minimizes the error according to the least squares criterion used by CART.

In Algorithm 5.5 we present the rule selection process in detail. From all the rules that form the ensemble, we select the best estimated rules within the top η margin rules according to the specified utility metric $\mathcal{U}$. Each rule is evaluated separately as an independent model. From this evaluation, we obtain a ranking of rules. If $0 < \eta \leq 1$ then it indicates the percentage of top performance rules to include in the final model. If $\eta > 1$ then it refers to the absolute number of top performance rules to include in the final model. At the end, the default rule is added to ensure that the final ensemble does a complete coverage of the input space.

5.5 Further Work

There are three main aspects that are worth to be studied further with the goal of improving this algorithm: (i) building a regression tree with an utility metric as preference criterion instead of the least squares criterion; (ii) making an interpretability analysis of the rules, and (iii) using a more effective rule selection technique. Next, we briefly present possible paths for each of these issues.

Algorithm 5.5 ubaRulesSel($\langle \mathcal{DS}, \phi \rangle, f_M, \mathcal{U}, \Lambda$): utility-based selection of regression rules.

Input: a regression problem described by a data set $\mathcal{DS} = \{\langle \mathrm{x}_i, y_i \rangle\}_{i=1}^{N}$ and a relevance function ϕ, an ensemble of regression rules $f_M(\mathrm{x})$, an utility metric $\mathcal{U}$ and a set of ensemble parameters Λ, where $\eta \in \Lambda$ indicates the margin of rules to select.

Output: $f^*(\mathrm{x})$ - Utility-based Regression Rules Model.

$R_M \leftarrow \{r_k(\mathrm{x})\}_{k=1}^{K}$ // initial set of rules contained in f_M
$r_0(.) = \overline{Y}$ // default rule
for $k \leftarrow 1$ **to** K **do**
  $R_k(\mathrm{x}) \leftarrow r_k(\mathrm{x}) \oplus r_0(.)$
  $\mathrm{u}_k \leftarrow \mathcal{U}(R_k(\mathrm{x}), \langle \mathcal{DS}, \phi \rangle)$ // estimate utility performance
end for
if $\eta < 1$ **then**
  $\eta \leftarrow \eta\, K$
else
  $\eta \leftarrow \min\{\eta, K\}$
end if
$o \leftarrow \mathsf{order}(\mathrm{u}, \mathit{decreasing} = \mathbf{true})$ // rank by utility in decreasing order
for $k \leftarrow 1$ **to** η **do**
  $r_k(\mathrm{x}) \leftarrow r_{o_k}(\mathrm{x})$ // update the final model
  $\hat{a}_k \leftarrow \hat{a}_{o_k}$
end for
$f^*(\mathrm{x}) \leftarrow \frac{\sum_{k=1}^{\eta} \hat{a}_k\, r_k(\mathrm{x})}{\sum_{k=1}^{\eta} \hat{a}_k} \oplus r_0(.)$ // add the default rule
return $f^*(\mathrm{x})$

Utility-based Regression Trees

One of the possible extensions of our system is to build regression trees that are not guided by the least squares criterion, but biased by our proposed utility metric. These trees will derive more interpretable and accurate rules for the domain user. To achieve this goal, a Maximal Utility criterion can be established. Let us assume that k is the constant that maximizes the expected utility value of the target value. In such context, the estimated utility of a node t would be defined as

$$Util(t) = \mathcal{U}(k_t, D_t) \tag{5.12}$$

where k_t is the constant that maximizes the expected value of the utility on the partition D_t of node t.

The utility of a split s over the node t could be estimated by a linear function f (e.g. the weighted average) of the utility of its left subnode t_L and right subnode t_R as follows,

$$Util(s,t) = f(Util(t_L), Util(t_R)) \tag{5.13}$$

The best split s^*, from the set of candidate splits S at a node t, would be the one that maximizes the utility estimative of t the most, i.e.

$$s^* = \underset{s \in S}{\text{argmax}}(Util(s,t) - Util(t)) \tag{5.14}$$

This proposal faces two challenges: definition of the constant k and computational efficiency. In what concerns the constant k, it should maximize the expected utility on the target variable. Given that the target variable Y is continuous, we know that its expected value, with respect to a set of instances $y \in \mathbb{R}$ drawn from a distribution D, is given by

$$E_{y \sim D}\ [y] = \int_{-\infty}^{+\infty} y\, f(y)\, dy \tag{5.15}$$

where f is the probability density function of the target variable Y.

In this context, our goal is that, with respect to a set of instances $\langle \mathrm{x}, y \rangle \in \mathcal{X} \times \mathcal{Y}$ drawn from a distribution D, k maximizes

$$E_{\langle \mathrm{x}, y \rangle \sim D}\ [\mathcal{U}(k,y)] = \int_{-\infty}^{+\infty} \mathcal{U}(k,y)\, f(y)\, dy \tag{5.16}$$

where $\mathcal{U}$ is an utility-based metric and f is the probability density function of the target variable Y. However, the utility metric $\mathcal{U}$ is highly dependent on the relevance function ϕ, a user-specified function, and on its bumps of relevance as described in Chapter 3, Section 3.3.2.3 - page 92. Hence, solving the above integral is not straightforward.

The other challenge of this proposal is its computational efficiency. In effect, the efficient growth of the least squares regression trees, such as CART, is based on a statistical approximation of the variance (Torgo, 1999). This allows a faster calculation of the mean value, the constant that minimizes the expected error on a given partition. As such, if another constant, different from the mean, is found to be the one that maximizes the expected utility, this "speed up" algorithm trick can no longer be used. This means that, unless any other statistical simplification is discovered, the utility estimate would have to be calculated for each new candidate split from scratch, instead of incrementally as it occurs in standard regression trees. For large data sets this would not be feasible. Nevertheless, we consider that, from the perspective of utility, better models will be possible to obtain.

A Possible Extra Interpretability Analysis

The theory as a whole, consisting in a number of rules with a number of conditions over a set of variables, can be subject to an interpretation analysis. Many rule systems, such as RuleFit (Friedman and Popescu, 2008), carry an analysis regarding the importance of each rule, their input variables importance, their interaction effects, etc. The joint contribution of all these factors about the rules express the interpretability of a theory.

As future work, we would like to incorporate an interpretability analysis into our rule evaluation scheme. Namely, we would like to examine the rules concerning its support on the training data (the number of examples it covers), its length (the number of conditions it has) and the inspect relevance ϕ of its region of influence on the target variable. We believe that a more informed selection process can then be performed and more interpretable models obtained.

A Better Selection Method for Rules

Further work on rule selection/combination stage can also be carried out. In particular, we can select the rules using more powerful techniques, which contribute more to the overall optimization of the utility metric. Namely, for the case of rare extreme values prediction, we believe that a selection of rules based on the AUC-PRIV of achievable PRIV curve, following a strategy similar to the ROCCER algorithm Prati and Flach (2005), would yield to smaller rule sets with compatible AUC-PRIV values.

5.6 Summary

In this chapter, we have proposed ubaRules, a new system to address regression tasks with non-uniform costs/benefits on the target variable. Though this type of problems have not been subject of many research among the data mining community, they occur in many real-world applications. Prediction of meteorological/ecological catastrophes, finance, prediction of important business related decisions, detection of frauds, intrusions on a network, are just some examples of applications that can find use in this type of models we proposed. Usually this type of applications are addressed by a classification approach, but that will not always fullfil the goals of the applications. Sometimes minor changes in the range of the continuous target variable can make a big difference for the domain expert, as many types of costs and/or benefits can be involved.

In this context, we have presented an utility-based regression rule ensemble system that aims to obtain interpretable and accurate models for the prediction of user-defined relevant values in the range of the continuous target variable. In the next chapter, we will experimentally evaluate and compare our system with standard regression systems for a set of real-world applications.

- 6 -

Empirical Analysis of ubaRules

"Why think? Why not try the experiment?"
John Hunter (1728 - 1793)

The goal of the utility-based rules ensemble system (ubaRules) is to effectively address regression tasks with non-uniform costs/benefits. Our aim is now to check through a series of experiments whether a set of variants of our system is able to overcome limitations of standard regression approaches for addressing such type of regression tasks.

6.1 Introduction

In Chapter 5 we have presented ubaRules, an utility-based regression rules system, whose main goal is to obtain accurate and interpretable predictions in the context of regression problems with non-uniform utility.

In this chapter, we experimentally evaluate ubaRules regarding its performance in terms of utility score, presented in Chapter 3, Section 3.3 (see page 79), and also in terms of the interpretability of the induced models. We start by describing the test domains used in this empirical analysis and the experimental and evaluation methodology. Next, in Section 6.3, we present the results of the experiments that compare our method with other existing methods in three different contexts. The first is designed to evaluate our system in the context of the prediction of rare extreme values; another is to inspect how effective our regression approach is in comparison to other known successful cost-sensitive classification methods; and, finally, one last context where we perform a comparison from the interpretability perspective. Throughout these experiments we also investigate the impact of some of our learning algorithm parameters.

6.2 Experimental Methodology

In this section, we describe the different experimental setups used in this empirical study. Our purpose is to test the hypothesis that ubaRules is the most suitable method for addressing regression tasks with non-uniform costs/benefits.

6.2.1 Test Domains

The data sets used in our experiments are all real-world applications. Some of them have been used and described in Chapters 3 and 4, while others are introduced and described here. All of them consist of non-uniform costs/benefits regression tasks for the prediction of rare extreme values. Hence, regarding each data set, in the absence of domain knowledge, the relevance function is defined based on the box plot of the target variable, following the methodology proposed in Chapter 4, Section 4.2.2 (see page 122). In terms of results presentation, we have aggregated the test domains into the three following groups:

- Miscellaneous domains (`MISC`): domains with the regression task of rare and extreme values prediction. This group includes the following data sets:
  - `NO2Emissions`: a data set with 500 examples and 8 attributes regarding climate conditions and traffic information (Aldrin and Haff, 2005), factors that influence the outdoor air pollution. The prediction task is to forecast the emissions of the nocive NO_2 in an urban area. To fulfill such goal, european environment directive guidelines are used to avoid high extreme NO_2 Emissions, as described in Chapter 3.
  - `ForestFires`: a data set formed by 2831 examples and 86 attributes. These attributes respect land morphology, land characteristics, social-demographic values and weather variables. The prediction task is to forecast the forest fires with the biggest impact, i.e. the forest fires whose percentage of burnt area is extremely high as presented in Chapter 4.
  - `Boston`: data set available in package `MASS` (Venables and Ripley, 2002) of R environment (R Development Core Team, 2010). It has 506 examples described by 14 variables regarding social-demographic aspects of Boston residential areas, such as per capita crime rate by town, pupil-teacher ratio by town, nitrogen oxides concentration, the proportion of blacks by town, among others. The prediction task is to predict the housing highest and extreme values in the residential areas of Boston.

- Stock Market (SM): this second group of problems includes 12 data sets concerning the task of trying to forecast future stock market returns. The 12 prediction tasks use data concerning the standard daily quotes for almost 20 years of four companies - International Business Machines (IBM), Coca-Cola (KO), Boeing (BA) and General Motors (GM). The target variables include predicting the 1, 3 and 5-days ahead returns of these four companies. The goal here is to make accurate predictions of the extreme and rare high (positive and negative) variations of the returns, as these yield to the most profitable trading actions (cf. Chapter 4). Each data set reports to the period from 1970-01-02 till 2008-07-11, in a total of 9725 daily sessions. Regarding the predictor variables, we have used an embedding of the previous 24 daily quotes.

- Harmful Algae Blooms (HAB): the final group of problems concerns the appearance of harmful algae blooms, a well known ecological problem in several european rivers. The 6 prediction tasks consist in forecasting these algae blooms, i.e. rare and extremely high concentration of certain species of micro-algae, which can be critical in terms of potential impact to public health. The data refers to a collection of samples made in river Douro in Porto, Portugal, during the period from 1998 to 2003. Each observation respects a biweekly period. We have created 6 different prediction tasks, one per each micro-algae we want to forecast, all with 131 observations and 35 attributes regarding physical-chemical and microbiological parameters of the water samples.

6.2.2 ubaRules Parameterization

In these experiments, and given that the goal of our test domains is to predict rare extreme values, we established F-measure with $\beta = 0.5$ for an event threshold $t_E = 1$ (cf. Chapter 4, Section 4.2.2 - page 122) to be the preference criterion of ubaRules. We have also used an even penalization factor for the costs, i.e. $p = 0.5$. We then have considered the following 8 variants of ubaRules. We have used both bagging (bagg) and boosting (boost) ensemble techniques to grow 100 and 200 trees. Into the final model, we included all the produced rules (All) and the top 90% best ranked rules (T0.9), according to utility estimates. For all the variants, and to bound the complexity of generated rules, we have considered an average maximum tree depth equal to 4.

These settings will not give us a complete overview of ubaRules behaviour regarding the test domains. Still, it is not our intention to tune ubaRules but to get a notion of where ubaRules "stands" compared to standard regression methods, as an approach to solve this type of regression tasks. Through these set of experiments, we aim to discover what aspects of ubaRules are worth of a deeper investigation.

6.2.3 The Modelling Tools

In this section we provide a brief overview of the different modelling techniques that are used as competitors to our system. The objective is to use a set of different approaches to the same set of regression problems. The selected competitors methods are intended to represent a sample of regression modelling techniques known by their effectiveness. All tools we have used, with the exception of the Rulefit system, are available in the R statistical environment (R Development Core Team, 2010).

- Regression Trees (cart, ubaCart): we use the `rpart` (Therneau et al., 2008) package functionalities to obtain the regression trees. The function `rpart` implements both classification and regression trees in a way similar to CART system (Breiman et al., 1984). The following post-pruning rules are used to obtain different regression trees:
  - cart: use the function `rpartXse` from the package **DMwR** (Torgo, 2010), as an interface to `rpart`, and establishing `se=1` to obtain 1-SE CART regression trees;
  - ubaCart: use the function `rpart` and establishing no post-pruning and then choosing the "best-sized" tree using utility as evaluation metric of the CART post-pruning rule, as explained in Chapter 5, Section 5.3.2, page 164.
- Random Forests (randomF): we use the R package `randomForest` (Liaw and Wiener, 2002) that implements Breiman's random forest algorithm (Breiman, 2001) for regression and classification. For random forests, we set the number of trees (parameter `ntree`) of the ensemble to 500.
- Multivariate adaptive regression splines (mars): we use package **earth** (Milborrow et al., 2010) of R, and establish the maximum degree of interaction with `degree=2` and the threshold for forward stepping with `thresh=0.01`.
- Support Vector Machines (svm): we use the `e1071` package (Dimitriadou et al., 2010), and in particular the `svm` function and set `cost=100` and `gamma=0.01`. The former is a constraints violation parameter, while the latter is a radial basis function kernel parameter.
- Rulefit (rulefit): we use Rulefit3$^{\text{tm}}$ [*1], which implements an interface with R software environment (R Development Core Team, 2010). We run rulefit with its default parameterization [*2] that assumes a maximum number of rules of 2000, the average tree size equal to 4 and mixed (rules and linear) model mode type.

[*1] `http://www-stat.stanford.edu/~jhf/r-rulefit/rulefit3/R_RuleFit3.html`

[*2] A complete description of the Rulefit parameterization can be found on its help manual page in `http://www-stat.stanford.edu/~jhf/r-rulefit/rulefit3/RuleFit_help.html`.

6.2.4 Evaluation Metrics

The results obtained by the models are evaluated by the same metric that was used as preference criterion to ubaRules, i.e. F-measure with $\beta = 0.5$ ($\text{Fm}_{\beta=0.5}$), considering an event threshold $t_E = 1$. Nevertheless, any other of our proposed utility-based performance metrics, such as AUC − PRIV, MAP11, BFM, could have been chosen to characterize the performance of the models for this type of applications.

6.2.5 The Experimental Settings

The prediction tasks used in our experiments can be divided in two different types of tasks: standard multiple regression tasks, and a second group where each case is associated with a time stamp, i.e. multiple regression tasks that were created from data that originally was a time series. In effect, in this second group the cases were generated by some form of aggregation of the original time series data at equally spaced time intervals.

In the first type of tasks we have the miscellaneous problems (`MISC`), where there is no time dependence between the instances. For these problems we have used stratified 10-fold cross-validation as experimental methodology. The use of the stratification is justified by trying to reproduce the same distribution of data, and thus the rare extreme values, at each of the 10 folds.

For the second type of tasks, based on multivariate time series data, which include the Stock Market (`SM`) and Harmful Algae Blooms (`HAB`) problems, we have used the same experimental setting presented in Chapter 4, Section 4.5.1 (see page 143) and Section 4.5.2 (see page 150).

In order to verify the statistical significance of the observed differences among the 14 compared methods, we have used two statistical tests. We apply the *Friedman* test to detect if there are differences in the compared methods across all the data sets that are included in the experiment. The procedure involves ranking the estimates obtained by each method for each data set considering all the methods. For each method we can then calculate its average ranking. Using this test we can check if the methods can be considered equivalent or not at some confidence level. Still, even if the *Friedman* test may indicate us that the methods are not equivalent, it tells us nothing about the ranking of the methods themselves in the current experiment. If we have a significant difference among the methods, we can apply the *post-hoc Bonferroni-Dunn* test to compare all methods to each other. Due to the great number of comparisons involved, the *post-hoc Bonferroni-Dunn* test adjusts the p-value to avoid the family-wise error rate (Demšar, 2006). This rate is the probability that one or more of the significance tests wrongly rejects a true null hypothesis in a series of tests.

This means that the test would signal two models as being significantly different when, in fact they are not. The *Bonferroni-Dunn* adjustment controls the family-wise error rate by dividing α (the significance level) by the number of comparisons made. With these two statistical tests, we will inspect the significant differences between all the 14 modelling techniques, with a 95% confidence level. In the following section, we present some of the results of these tests in tables and Critical Difference (CD) Diagrams as proposed by Demšar (2006). In Appendix B we provide the complete set of results.

6.3 Experimental Results

In this section we report the obtained results according to the previously specified experimental settings. Our main goals are to be able to answer the following three questions regarding the behavior of ubaRules: (i) does it provides any utility improvement when compared to other standard regression systems in the context of our target applications? (ii) how does it compare, in terms of utility, to the alternative of addressing our tasks through a cost-sensitive classification approach? (iii) finally, when compared with another rule system, how large is the number of rules selected by our ensemble?

6.3.1 Utility Evaluation

In this section, we present an overall analysis of the utility performance estimates obtained by all the selected modelling techniques over the data sets that share the same experimental setting. For better visualization purposes, we do not always include all the 8 ubaRules variants in the Critical Difference (CD) diagrams. Still, the results are obtained through a statistical analysis that includes all the 8 variants of ubaRules plus the 6 other alternative modelling tools mentioned above, as competitors of ubaRules. The complete set of results is available on Appendix B.

In the context of miscellaneous data domains (`MISC`), the *Friedman* test and *post-hoc Bonferroni-Dunn* test did not report any statistical difference between all the 14 compared methods. This is illustrated by the Critical Difference (CD) diagram in Figure 6.1.

Each method is represented by a symbol, and plotted on X-axis at its respective $\mathrm{Fm}_{\beta=0.5}$-based average ranking position. The higher the $\mathrm{Fm}_{\beta=0.5}$ estimate, the better is the ranked position. The Y-axis has no meaning. These diagrams also present the information on the statistical significance regarding every pairwise comparison between methods, which means that for each method we find several lines, one for each pairwise comparison with the remaining methods. The lines linking any

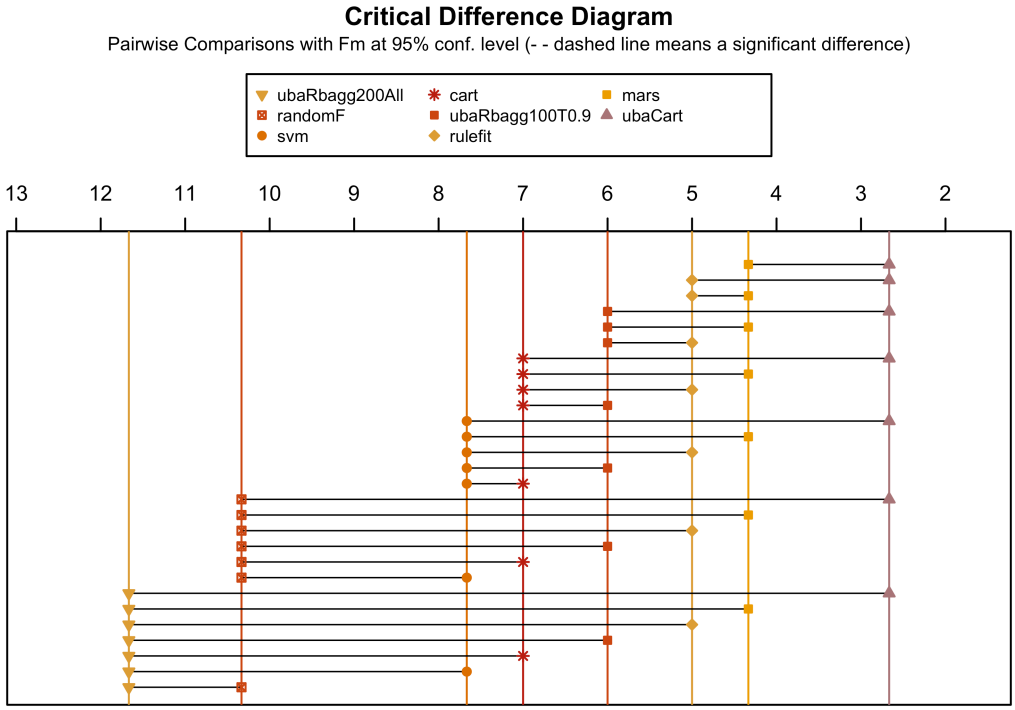


Figure 6.1: Critical Difference (CD) diagram obtained with $\text{Fm}_{\beta=0.5}$ estimates of all competitors within miscellaneous domains data (`MISC`).

two symbols indicate the statistical significance of the respective pairwise comparison. A dashed line connecting two symbols has the meaning that according to the *post-hoc Bonferroni-Dunn* test, the methods are significantly different with a 95% confidence level. Bold lines indicate that the difference in average ranking is not statistically significant at the same confidence level. An ideal performance would be a method whose symbols are at the right most position of the graph (lowest average ranking), and are connected only by dashed lines to every other method (all pairwise comparisons are statistically significant).

As we have mentioned, regarding the miscellaneous domains setup, no significant differences were registered among the average ranking position of the tested models. Still, it is possible to observe from the CD diagram on Figure 6.1 that ubaCart, the modelling technique that selects the best utility post-pruned regression tree, occupies the leading position. The models ubaRbagg100T0.9 and ubaRbagg200All are the best and the worst ranked variants of our system on this setup. In these conditions, according to the utility estimates provided by $\text{Fm}_{\beta=0.5}$, one can say that increasing the number of iterations of ubaCart trees is not having the expected gain in the performance of ubaRules. In effect, ubaRules does not achieve significant improvements when compared to standard regression systems. We believe this is due to the fact of the rules of ubaRules being produced by a CART (Breiman et al., 1984) tree, which minimizes the mean squared error. However, these results are only indicative as they have no statistical significance The overall set of $\text{Fm}_{\beta=0.5}$ estimates and

average ranking performance obtained by each model for this experimental setup is available on Appendix B, Table B.1.

Regarding the Stock Market domain, the *Friedman* test and *post-hoc Bonferroni-Dunn* test reported some statistical significant differences between the tested modelling techniques. In Table 6.1 we present the pairwise comparisons of the models for which the average ranking position is different with a confidence level of 95%, i.e. with p-value < 0.05.

Table 6.1: Significant differences in pairwise comparisons of $\text{Fm}_{\beta=0.5}$ estimates of modelling techniques in Stock Market Data. The bullet (•) signs, with a 95% confidence level, the best ranked modelling technique.

Pairwise Comparison over SM data				$\text{Fm}_{\beta=0.5}$ p-values
cart		svm	•	**0.0278**
svm	•	rulefit		**0.0278**

From Table 6.1, we see that svm achieves a ranking position with respect to $\text{Fm}_{\beta=0.5}$ performance, that is significantly better than the ranking positions obtained by cart and rulefit. In effect, based on $\text{Fm}_{\beta=0.5}$ performance estimates, no other statistical significant differences were identified among the ranking positions of all the remaining competitors as illustrated by the CD diagram (cf. Figure 6.2). Moreover, we see that svm has the leading ranking position followed by ubaCart, which again obtains a good overall average ranking positioning even though not a statistical significant one. These two models (svm and ubaCart) appear isolated in front of all the remaining models. The cart and rulefit share the same last ranking position for this task. The remaining modelling techniques have equivalent ranking performances. To illustrate the positioning ubaRules in the context of all the competitors, we have selected the variants ubaRbagg200All and ubaRboost200T0.9, the variants that achieve the lowest and highest average ranking, respectively.

For the Harmful Algae Blooms domain, the conducted statistical analysis reported no significant differences. In Figure 6.3, we present the critical difference (CD) diagram drawn from the obtained results over Harmful Algae Blooms data. For this set of experiments, the svm appears at the leading ranking position followed ubaRboost200All, the best ranked ubaRules variant in this setup. At the worst ranking postions, we find modelling techniques like cart and also ubaCart. Even though these results are not significant, they are consistent, to what we have obtained in the first set of experiments over this same set of data (cf. Chapter 4, Section 4.5.2 on page 150). On those set of results svm appeared among the good candidates for modelling and standard regression trees among the worst. Even so, our worst ranked variant, ubaRbagg200T0.9, is closer to more competitive models such as randomF or mars.

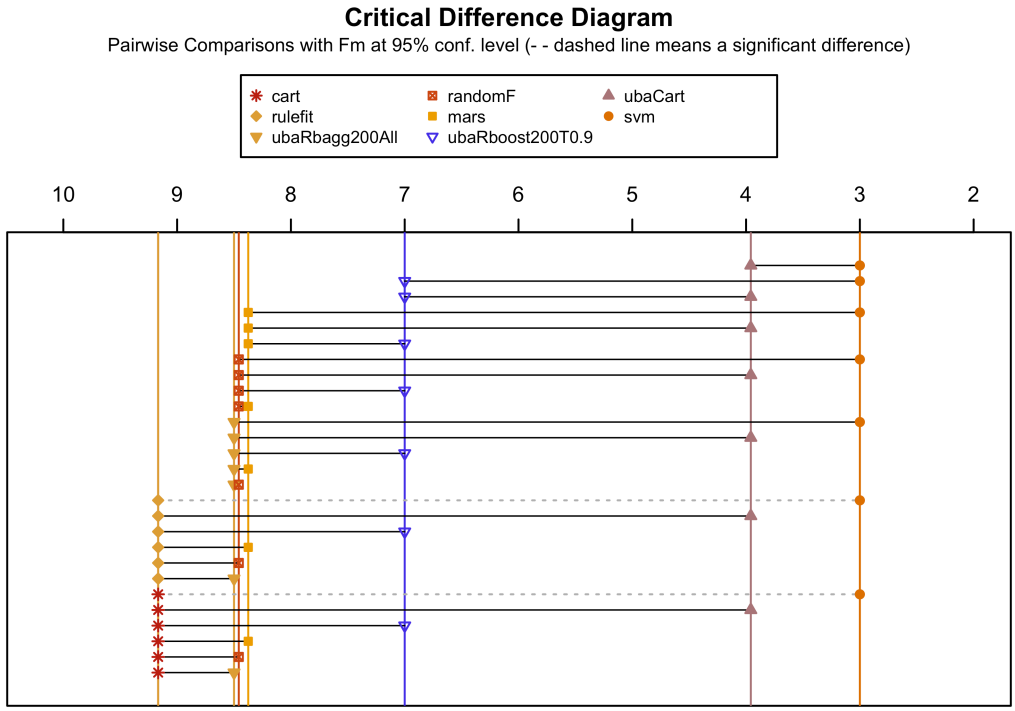


Figure 6.2: Critical Difference (CD) diagram obtained with $\text{Fm}_{\beta=0.5}$ estimates of all competitors within Stock Market data (`SM`).

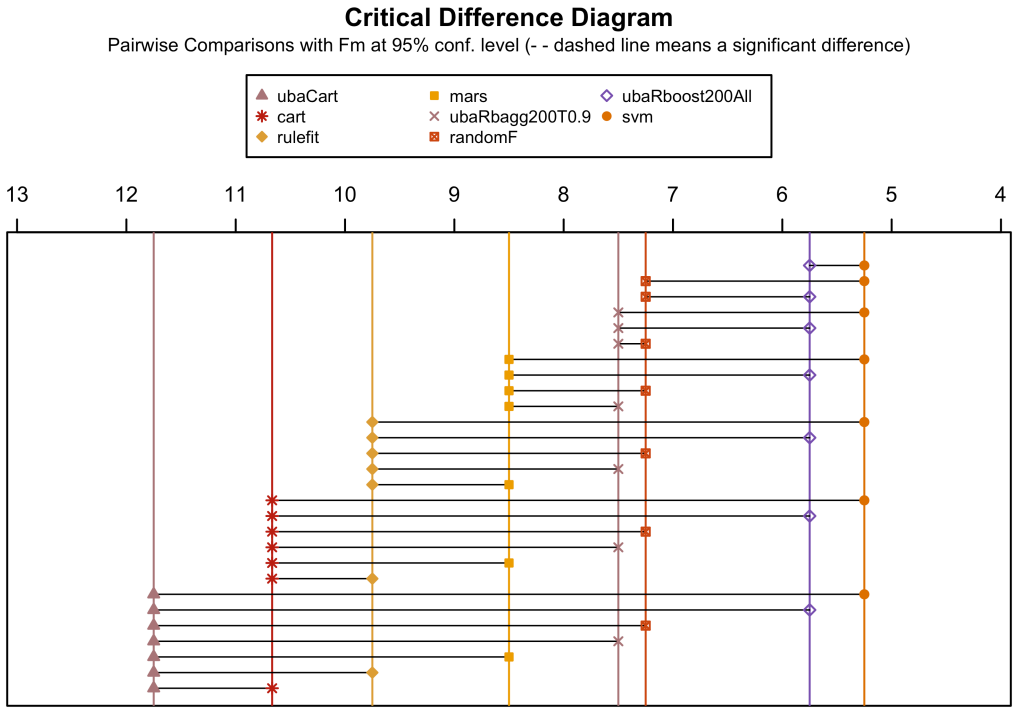


Figure 6.3: Critical Difference (CD) diagram obtained with $\text{Fm}_{\beta=0.5}$ estimates of all competitors within Harmful Algae Blooms data (`HAB`).

6.3.1.1 Discussion

We have performed an utility evaluation of our proposed system, in comparison to other standard regression methods. The results of the overall tried variants of ubaRules regarding the $\mathrm{Fm}_{\beta=0.5}$ performance estimates (cf. Appendix B, from Table B.1 to Table B.3) were not as good as we would expect. Standard regression methods, which optimize standard error metrics, achieve $\mathrm{Fm}_{\beta=0.5}$ estimates comparable to ubaRules. Nevertheless, even though ubaRules uses utility as preference criterion in the selection of best-sized tree and on the pruning of rules, the rules are themselves grown by standard CART (Breiman et al., 1984) trees. Thus, they are biased by the standard error. The experiments show that ubaRules performance is still instable and that there are still open questions. Namely, for most of the domains, boosting did not get better results than bagging, the number of trees seems to vary from domain to domain. Regarding the selection of rules, more tuning is needed.

6.3.2 Regression vs. Classification

Along this thesis, we have claimed that the existing cost-sensitive classification approaches are not suitable to address our target applications. In this section, we provide an experimental evidence of the comparison between our proposed regression system and other cost-sensitive classification modelling techniques. Our goal is to check whether the use of a probabilistic classifier with an appropriate cost-benefit matrix would not achieve the same effect as our utility-based framework.

In this context, we define the bumps of relevance of the target continuous variable to be the bins that constitute the classes for the classification task. Given that the prediction task in this experimental study is the prediction of rare and extreme values, we will have a two-class problem if there is only high or low extremes; or a three-class problem, if both type of extremes exist.

Each bin is assigned the median target value of the partition it represents. Hence, to the instance i is assigned the class c_i by the following discretization:

$$c_i = \begin{cases} \widetilde{Y_L}, & \text{if } y_i \in Y_L := \{y \in Y \,|\, y \leq Q_1 - 1.5\, IQR\}; \\ \widetilde{Y_H}, & \text{if } y_i \in Y_H := \{y \in Y \,|\, y \geq Q_3 + 1.5\, IQR\}; \\ \widetilde{Y_I}, & \text{otherwise.} \end{cases} \tag{6.1}$$

where $\widetilde{Y}$ represents the median of the values in Y, Y_L defines the set of extreme low values, Y_H defines the set of extreme high values, having that IQR is the interquartile range, Q_1 and Q_3 are the first and the third quartiles of the target variable Y.

These are cost-sensitive applications and thus we have defined the following cost/benefit classification matrix,

$$b_{ij} = \begin{cases} \max\{|i-1|^2, |i-n|^2\} & \text{if } i = j; \\ -|i-j|^2, & \text{otherwise.} \end{cases} \tag{6.2}$$

where n is the number of classes.

As probability classifiers, we used the classification ability provided by the functions mentioned in Section 6.2.3, which implement Support Vector Machines and Random Forests. We set them with the same parameterization referred before, and give additional information about the class priors following the discretization process mentioned above. The predictions are made, with the goal of maximizing the cost-benefit matrix.

Figures 6.4, 6.6 and 6.5 present the critical difference (CD) diagrams obtained in each domain, `MISC`, `HAB` and `SM`, with the comparison between these cost-sensitive classification approaches and the variants of ubaRules. Detailed results on this statistical analysis are available on Appendix B, from Table B.4 to Table B.6.

As it is possible to observe in all the 3 experimental setups, and according to $\text{Fm}_{\beta=0.5}$ estimates, all the tested variants of our system ubaRules did not exhibit an average performance significantly better than the two classification techniques. Still, it is possible to identify some variants of ubaRules in the two leading positions. Namely, ubaRbagg100All and ubaRbagg100T0.9 in the miscellaneous domain data; ubaRboost100All and ubaRboost200T0.9 in the Stock Market data; and all the tested ubaRules variants in the Harmful Algae Blooms data. Even though this is not observed in a significant way, it is important to notice that svm.class are randomF.class are two powerful and *overly tested* modelling techniques. Hence, a cost-sensitive classification made by such techniques can represent serious competition for our system.

Moreover, according to the above defined discretization process the target class matches exactly the target event cases, i.e. the cases that according to the utility metric $\text{Fm}_{\beta=0.5}$ with event threshold $t_E = 1$ have maximum utility. This puts the two classification approaches and our system in similar circumstances regarding the target variable.

Still, we claim that the main advantage of ubaRules is the smoothness of the relevance functions it uses, when compared to the roughness of the artificially formed bins for the classification approaches.

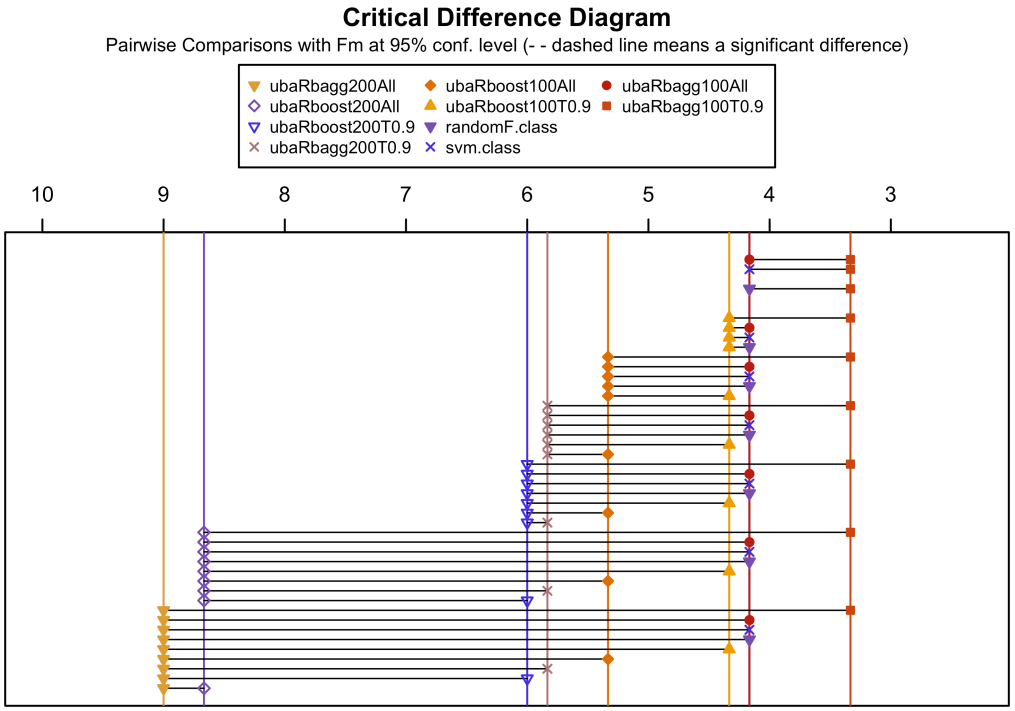


Figure 6.4: Critical Difference (CD) diagram obtained with $\text{Fm}_{\beta=0.5}$ estimates of classification competitors within miscellaneous domain data (`MISC`).

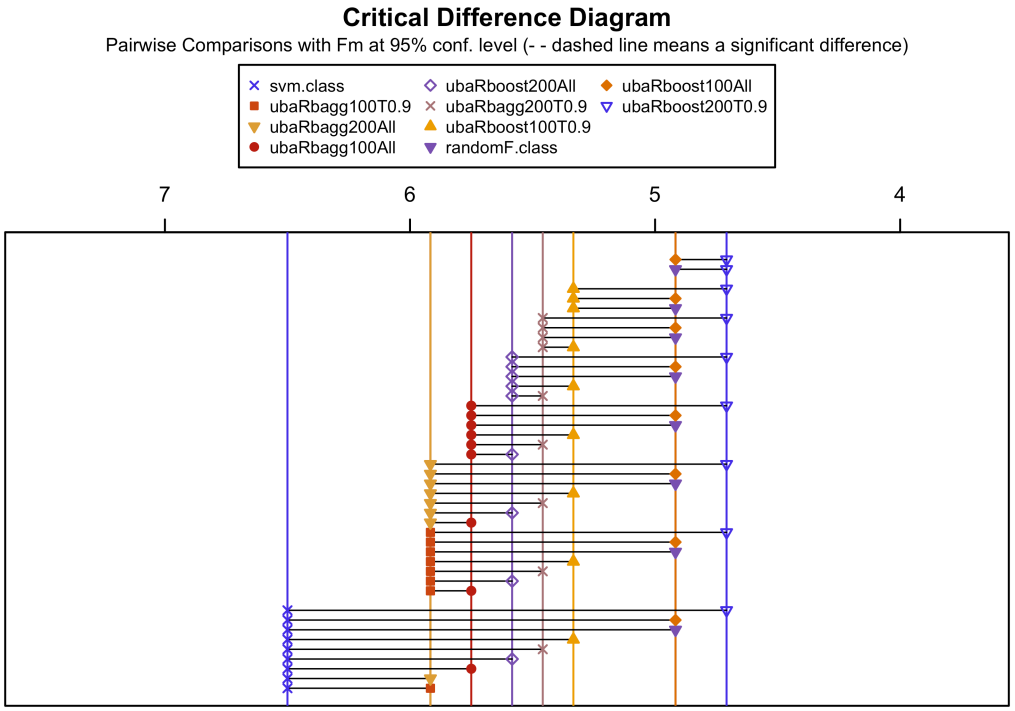


Figure 6.5: Critical Difference (CD) diagram obtained with $\text{Fm}_{\beta=0.5}$ estimates of classification competitors within Stock Market data (`SM`).

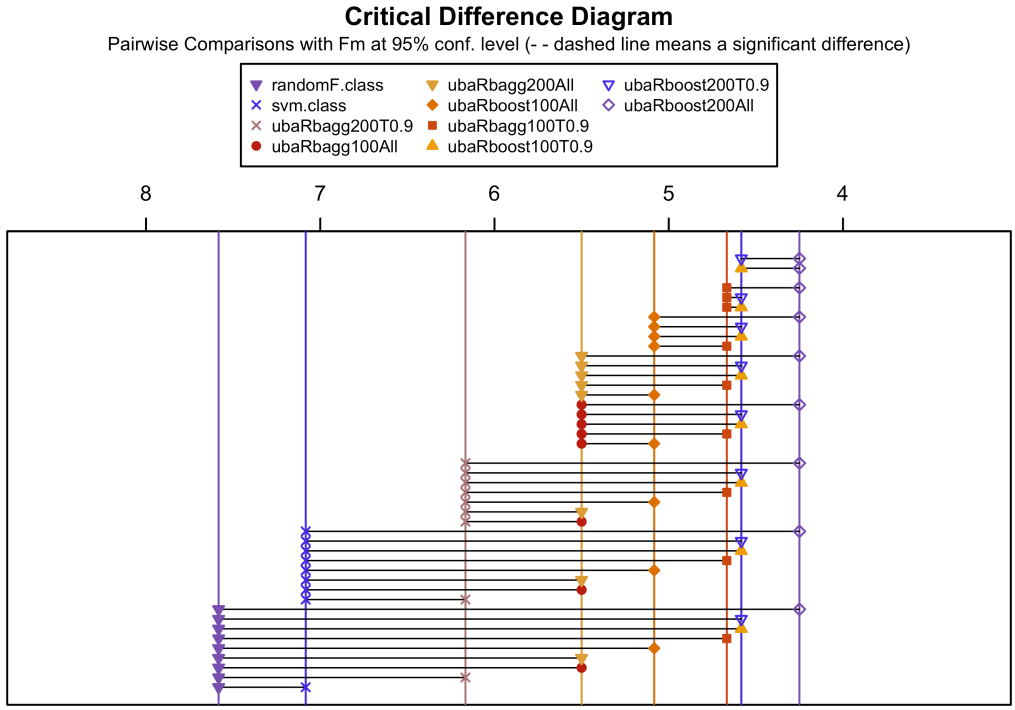


Figure 6.6: Critical Difference (CD) diagram obtained with $\text{Fm}_{\beta=0.5}$ estimates of classification competitors within Harmful Algae Blooms data (`HAB`).

6.3.3 Interpretability Issues

The main advantage of rules is their high interpretability. However, if the number of rules becomes too large, the interpretability of the rules ensemble as a theory may be questionable. Some authors (Dembczynski et al., 2010; Friedman and Popescu, 2008) argue that by assigning each rule a degree of interestingness or importance, the ensemble as whole can still be used in a highly interpretable manner. Regarding our ubaRules system, the same problem of interpretability of the ensemble arises. Thus, we have carried a set of pairwise comparisons between the variants of our system and rulefit (Friedman and Popescu, 2008) with respect to the number of rules in the ensemble. Still, we must point to the fact that ubaRules is not as powerful as rulefit on the rule selection phase. ubaRules does not use any interpretability analysis for this purpose. It only selects the top ranking rules according to some utility metric performance estimate.

We have performed a statistical analysis regarding the estimated number of rules generated by rulefit and all the 8 experimented variants of ubaRules. In these setting, the fewer is the average number of rules, the better it is. The statistical significance is asserted by *Friedman* test and then by *post-hoc Bonferroni-Dunn* test with a confidence level of 95%. All the results obtained from this analysis are available on Appendix B, from Table B.7 to Table B.9.

The critical difference (CD) diagram resulting from the analysis of the number of rules for miscellaneous domains data is shown in Figure 6.7. There is no statistical significance reported among the tested variants of ubaRules and rulefit. This means they all generate rule ensembles of equivalent size, with rulefit generating the bigger rule sets. Moreover, we can notice that the size of the rule ensembles generated by ubaRules with bagging and boosting is, on average, the same.

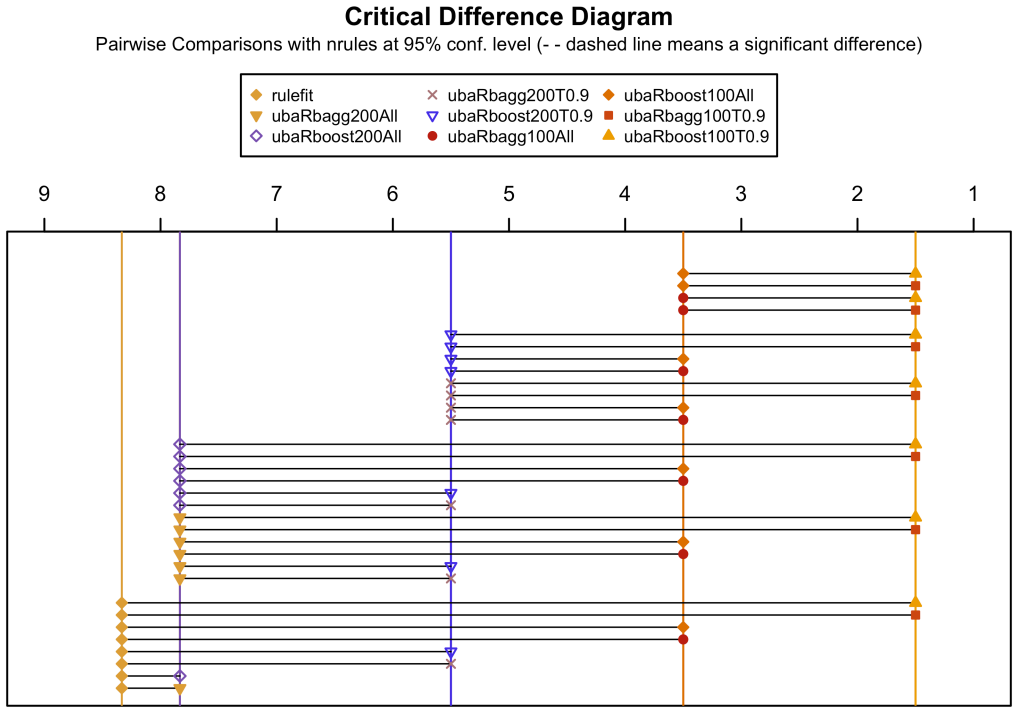


Figure 6.7: Critical Difference (CD) diagram obtained with rule set size estimates of rulefit and ubaRules variants within miscellaneous domain data (MISC).

The same type of analysis was carried over Stocks Market data. Still regarding this setup, statistical differences were reported. Table 6.2 presents the significant differences obtained for Stock Market data domain. The critical difference (CD) diagram resulting from this analysis is shown in Figure 6.8. In this setup, rulefit jointly with ubaRbagg100T0.9 and ubaRboost100T0.9 generate significant smaller rule sets. However, it is important to notice that the significance associated to rulefit is the result of void rules set for most of the data sets (cf. Table B.9 on Appendix B). This means that rulefit was unable to find a rule within the given setting. That also justifies its equivalent performance to cart shown on Figure 6.2. However, this is not the case of ubaRules variants, which were always able to generate rules. In particular, when running 200 iterations ubaRules generated significantly bigger rules ensembles when compared to ubaRbagg100T0.9 and ubaRboost100T0.9. Nevertheless, these two later variants are not among the best performing modelling techniques (cf. Figure 6.2), which leads us to conclude that the number of rules is not enough.

Table 6.2: Significant differences in pairwise comparisons of estimates of rule set sizes in Stock Market data. The bullet (•) signs, with a 95% confidence level, the best ranked modelling technique.

Pairwise Comparison over SM data			Nr. Rules p-values
rulefit	•	ubaRbagg200All	**0.0000**
ubaRbagg100All	•	ubaRbagg200All	**0.0094**
ubaRbagg100T0.9	•	ubaRbagg200All	**0.0000**
ubaRboost100All	•	ubaRbagg200All	**0.0094**
ubaRboost100T0.9	•	ubaRbagg200All	**0.0000**
rulefit	•	ubaRbagg200T0.9	**0.0028**
ubaRbagg100T0.9	•	ubaRbagg200T0.9	**0.0094**
ubaRboost100T0.9	•	ubaRbagg200T0.9	**0.0094**
rulefit	•	ubaRboost200All	**0.0000**
ubaRbagg100All	•	ubaRboost200All	**0.0094**
ubaRbagg100T0.9	•	ubaRboost200All	**0.0000**
ubaRboost100All	•	ubaRboost200All	**0.0094**
ubaRboost100T0.9	•	ubaRboost200All	**0.0000**
rulefit	•	ubaRboost200T0.9	**0.0028**
ubaRbagg100T0.9	•	ubaRboost200T0.9	**0.0094**
ubaRboost100T0.9	•	ubaRboost200T0.9	**0.0094**

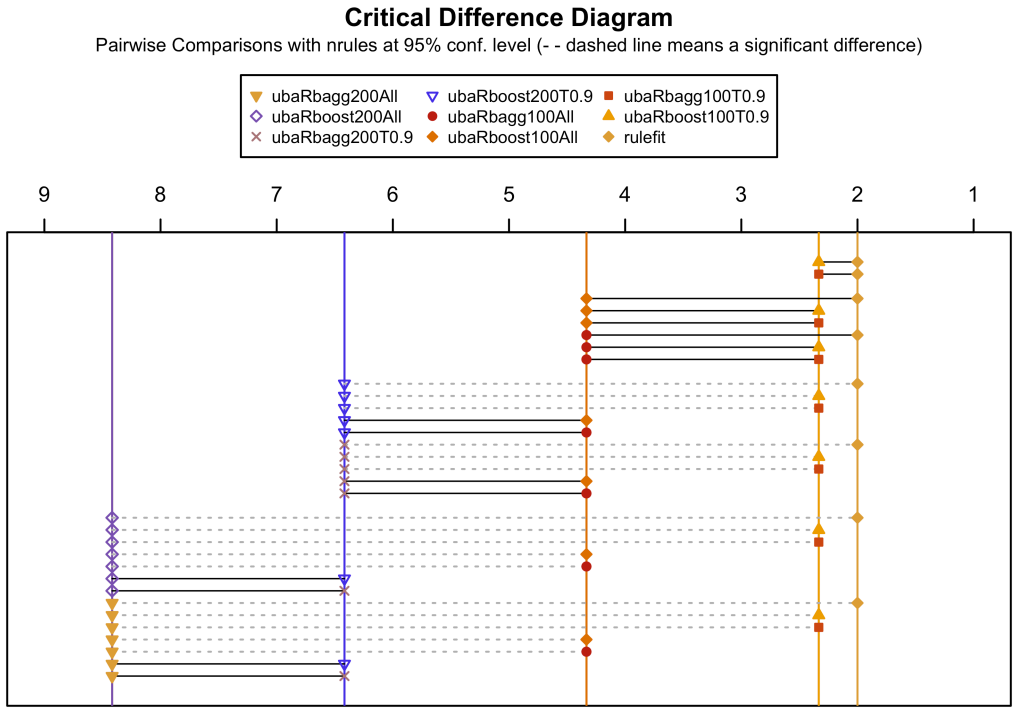


Figure 6.8: Critical Difference (CD) diagram obtained with rule set size estimates of rulefit and ubaRules variants within Stock Market data (SM).

In what concerns Harmful Algae Blooms data, the *post-hoc test* indicate us that ubaRbagg100All and ubaRbagg100T0.9 form significantly smaller rules sets when compared to rulefit, and ubaRbagg100T0.9 and ubaRboost100T0.9 also form significantly smaller rules sets when compared to ubaRbagg200All and ubaRboost200All (cf. Table 6.3). Nevertheless, and similarly to what we have observed in the Stock Market setup, none of these smaller rule ensembles are among the best performing models (cf. Figure 6.3). This give us indication more rules are necessary.

Table 6.3: Significant differences in pairwise comparisons of estimates of rules sets sizes in Harmful Algae Blooms. The bullet (•) signs, with a 95% confidence level, the best ranked modelling technique.

Pairwise Comparison over HAB data				Nr. Rules p-values
rulefit		ubaRbagg100All	•	**0.0182**
rulefit		ubaRbagg100T0.9	•	**0.0001**
rulefit		ubaRboost100All	•	**0.0182**
rulefit		ubaRboost100T0.9	•	**0.0001**
ubaRbagg100T0.9	•	ubaRbagg200All		**0.0053**
ubaRboost100T0.9	•	ubaRbagg200All		**0.0053**
ubaRbagg100T0.9	•	ubaRboost200All		**0.0053**
ubaRboost100T0.9	•	ubaRboost200All		**0.0053**

The CD diagram shown in Figure 6.9 illustrates the overall average ranking obtained by all tested ubaRules variants and rulefit. Once again, bagging and boosting of ubaRules generate the equivalent rule set sizes.

All these results confirm that ubaRules performance estimates would benefit from increasing the number of rules by both generating more trees and performing to a more careful rule pruning. Another aspect to further invest consists on the reason why boosting and bagging approaches do not differ. They are most of the times equivalent in the number of generated rules and estimated performance.

6.4 Conclusions and Further Work

In this chapter we have presented an empirical experimental study of our new proposed system ubaRules. The experiments have focused on real world applications where the goal is to predict rare and extreme values. Namely, we have used data concerning outdoor air pollution, forest fires, stock market, harmful algae blooms, among others. Some variants of our system and other competitive standard regression methods (e.g. cart, support vector machines, random forests, mars) were used in this experimental study.

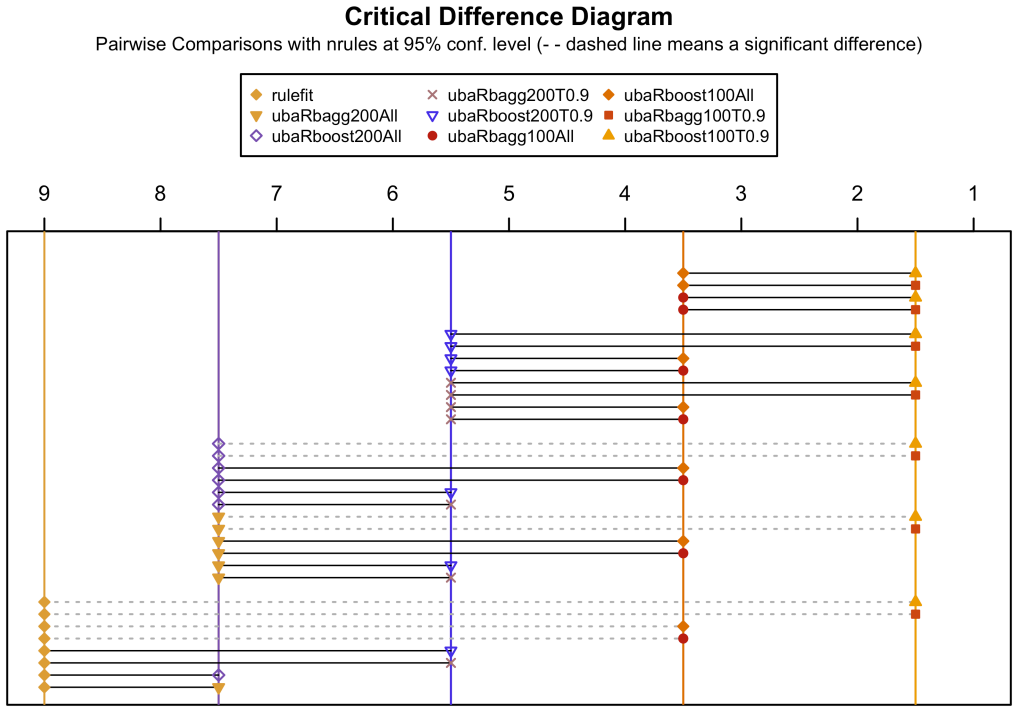


Figure 6.9: Critical Difference (CD) diagram obtained with rule set size estimates of rulefit and ubaRules variants within Harmful Algae Blooms data (HAB).

The overall set of results concerning ubaRules were not statistically significant. Still, according to $Fm_{\beta=0.5}$ estimates, ubaRules obtained results that are comparable to some of the most competitive techniques (e.g. support vector machines, mars). Moreover, these results concern specially hard prediction tasks, where other standard regression trees modelling techniques (e.g. cart) have difficulties. Still, a further investigation on ubaRules seem to be necessary to improve the results.

ubaRules is a modelling tool that addresses regression tasks allowing the specification of benefit/costs associated to different ranges of the target continuous variable. This information is incorporated as preference criterion of ubaRules that produces regression rules to fulfill the benefits/costs specifications. In this context, our proposed system is only a first approach to regression problems with non-uniform benefits/costs across the target variable domain.

- 7 -

Conclusion

"Now this is not the end. It is not even the beginning of the end.
But it is, perhaps, the end of the beginning."
Winston Churchill (1874 - 1965)

Addressing cost-sensitive applications involves two major issues: i) defining proper evaluation metrics to correctly assert the merits of alternative models given the application preference biases; and ii) defining learning strategies to better bias the models towards these goals. These two issues have been throughly addressed within classification problems. However, they have been essentially ignored in research on regression. Throughout this PhD work we have proposed an utility-based methodology that can cope with costs and benefits in regression problems. Based on this methodology we have derived a series of evaluation metrics that correctly assert the merits of prediction models in the context of these applications. Moreover, we have proposed a regression rule ensemble learning system that incorporates our utility-based metrics as preference criterion. Finally, we have carried out an extensive experimental evaluation of this system on different real world applications.

7.1 Summary

Real world applications have been widely addressed in classification throughout cost-sensitive techniques. In such type of applications, domains are characterized by having different utility (benefits or costs) according to the predicted and true class. We have argued that the same type of problems appear in regression problems, where a different benefit or cost should be awarded depending on the true and predicted values.

In this thesis we have addressed regression problems with non-uniform utility. The main goals of our study were: i) increasing the awareness of the research community for these important

tasks and in general to cost-sensitive regression; ii) exposing the risks of using standard regression evaluation metrics on cost sensitive applications; iii) proposing a new utility-based evaluation methodology that can cope with non-uniform regression tasks; iv) extending the utility-based evaluation methodology to metrics specially suited to handle rarity, namely for the prediction of rare extreme values of a continuous variable; v) developing an utility-based rule ensemble regression system that could cope with such evaluation requirements vi) providing empirical evidence on the advantages of the proposed system.

7.1.1 Contributions

This thesis has produced the following main contributions:

- **An evaluation methodology based on an utility function that can be used to in regression tasks with non-uniform costs/benefits.**
  We have shown that there is a set of regression problems with non-uniform costs/benefits, that are not properly addressed by standard regression models. These models try to optimize standard evaluation metrics used in regression tasks. We have demonstrated the inadequacy of these measures for these regression problems and have proposed a new evaluation framework based on the notion of utility. Contrary to standard regression metrics, we do not assume that the relevance of the continuous target variable is constant. Instead, we proposed to express it by a continuous function representing the domain knowledge on what is important for the end-user. Our methodology assigns an utility score to any prediction based on the relevance of both the true and predicted values and on the loss of the prediction.

- **Prediction of rare and extreme values through utility.**
  Decisions on trading actions based on oscillations of prices, or acting upon some ecological catastrophe, are two examples of critical decisions, which are typically related to the prediction of rare and extreme values of a continuous variable whose range of values define some target events. In these situations it is particularly important to get accurate predictions at the rare extremes values of the target variable, so that proper actions can be carried out. On these event-based applications evaluation measures like precision, recall and ranking measures are preferable. We have derived the equivalent of these measures for regression problems by incorporating the notion of utility of the predictions into both decision-making and ranking analysis measures.

- **ubaRules.**
  Based on the proposed utility-based metrics, we developed a new rule ensemble learning method, that tries to optimize our proposed utility-based criterion. ubaRules is a regression

rule ensemble learner biased by an utility metric as search heuristic. Its main goal is to obtain models that are both accurate from the utility perspective and also interpretable for the end-user.

7.2 Future Research Directions

Utility-based data mining is a relatively new field within *data mining* research. The main topic of this thesis, utility-based regression, is even more recent and thus there is a large space for improvements. As with any new approach to a problem, several aspects are still subject of further studies and possible improvements and/or changes.

In particular, the use of the proposed evaluation methodology as a mean to bias the search carried out by the modelling techniques should be further investigated. The robustness of the proposed utility-based evaluation should be confirmed by running further tests on the new methodology, namely regarding the specification of the relevance function. It would be interesting to see how our utility-based evaluation principle would affect the performance of classification or clustering techniques. We believe that the extension of our utility-based regression analysis to classification would have a great impact in data mining community. In effect, as cost-sensitive applications are standardly defined in classification, this would represent a confirmation of the effectiveness of our approach in comparison to the most well known approaches.

The proposed system ubaRules takes a "loose" approach to bias the models towards utility since it only carries out the incorporation of the utility metric at later stages of the learning process. This was mainly done for efficiency reasons. It is however clear that a tighter integration should lead to better rules, whose tests are chosen from the perspective of utility and not mean square error as it is the case of the rules used by ubaRules. Further work should clearly investigate these possibilities by addressing the computational efficiency questions raised by the use of the utility-based metrics.

Within the context of the current approach taken by ubaRules, there is clearly space for the rule fitting step and for a deeper investigation on its overall performance that is far from optimal as shown in Chapter 6. Moreover, the implementation of this system could also be improved in terms of efficiency, allowing its use on very large data sets.

- A -

Results of the Experiments in Stock Market Forecast

In this appendix, we present, by means of graphics, the complete set of results obtained from the experiments conducted for the Stock Market forecast problem presented in Chapter 4, Section 4.5.1 (see page 143). The following set of figures show the ranking of the 34 model variants that were considered for each of the 12 prediction tasks that describe this problem, according to the standard error MAD and to Fm with $\beta = 0.5$ based on our proposed definition of utility.

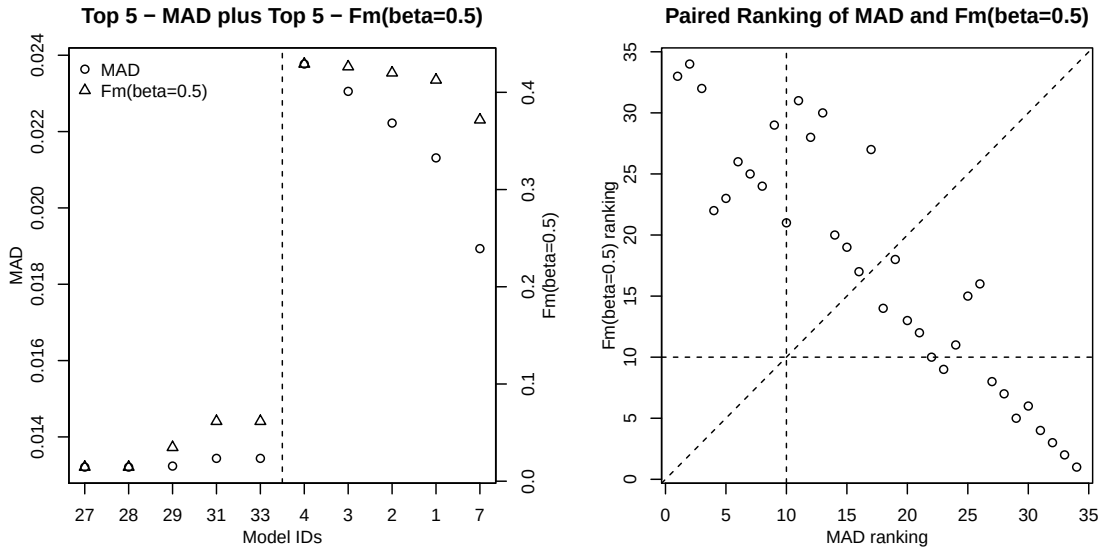


Figure A.1: The results for 1-days returns of International Business Machines (IBM).

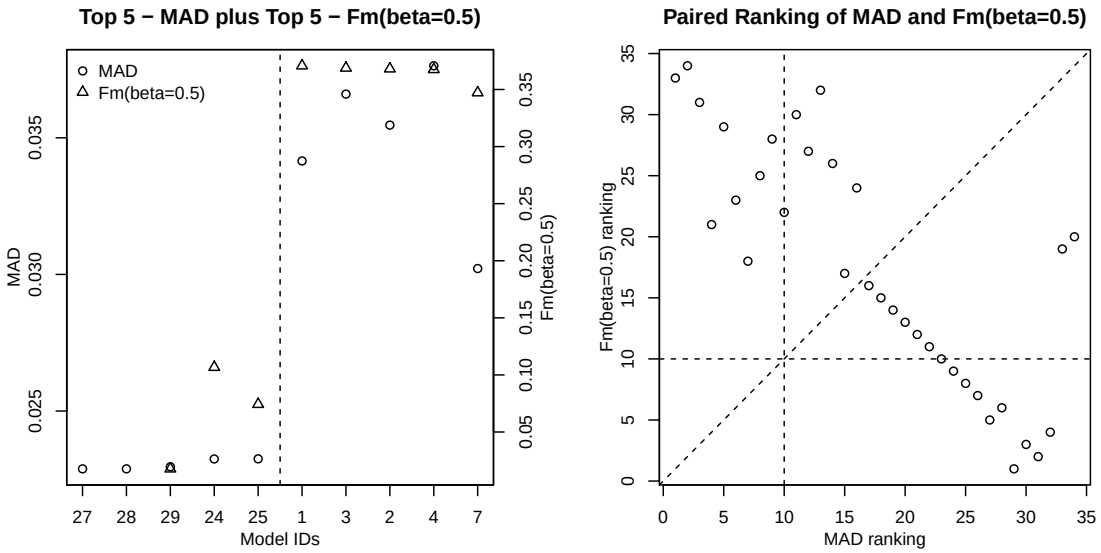


Figure A.2: The results for 3-days returns of International Business Machines (IBM).

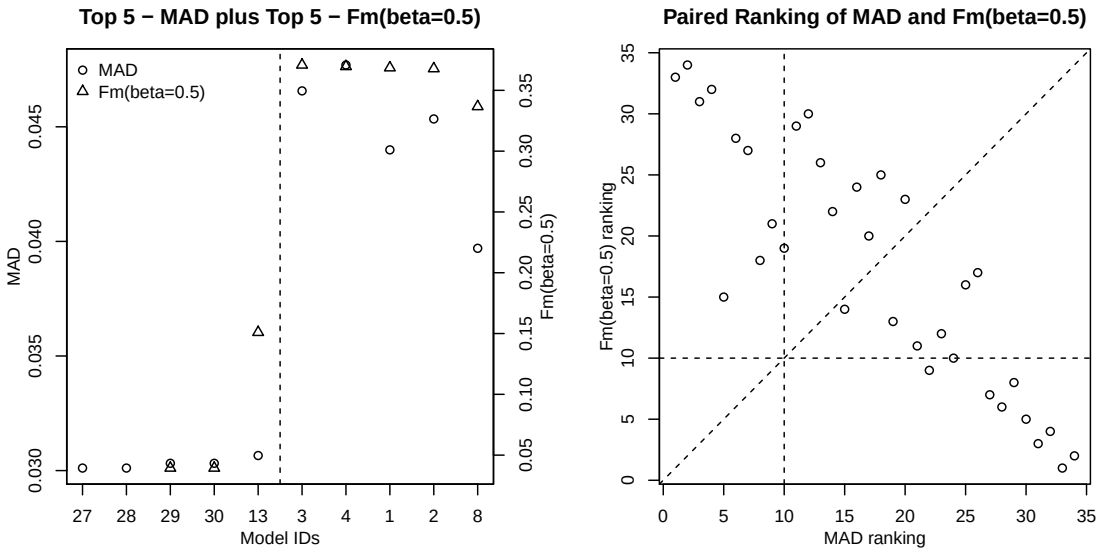


Figure A.3: The results for 5-days returns of International Business Machines (IBM).

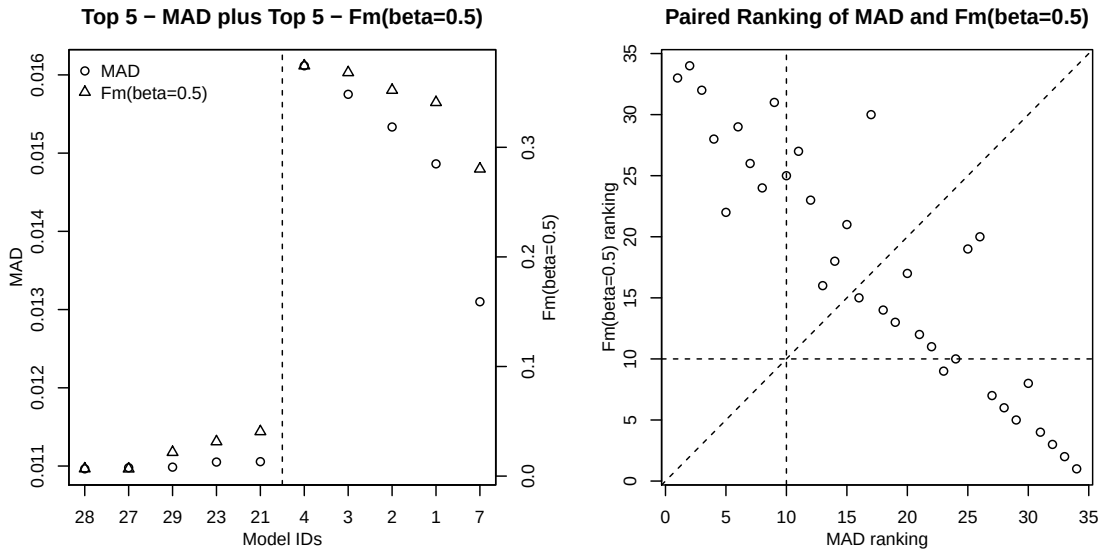


Figure A.4: The results for 1-days returns of Coca-Cola (KO).

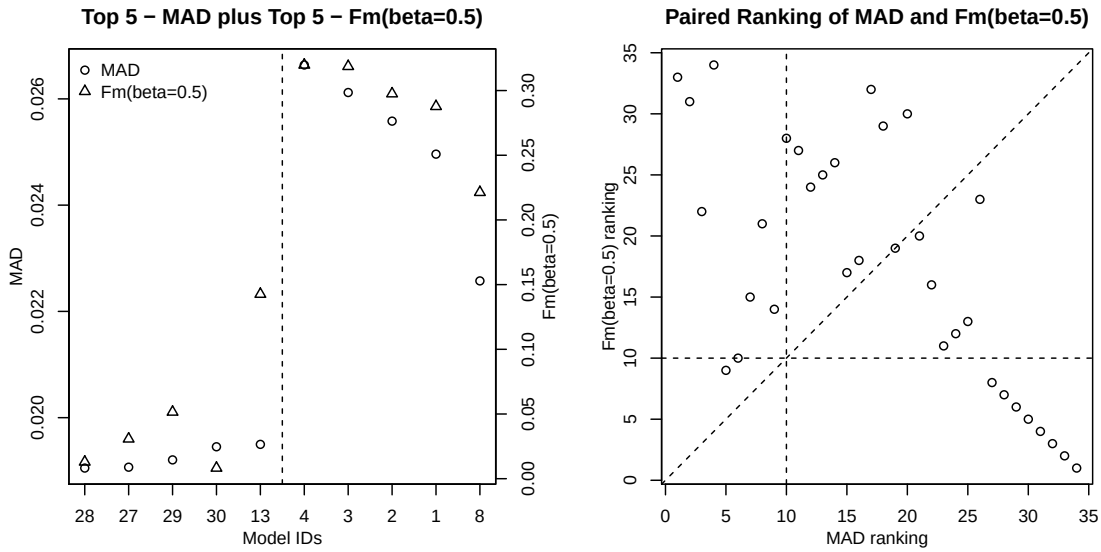


Figure A.5: The results for 3-days returns of Coca-Cola (KO).

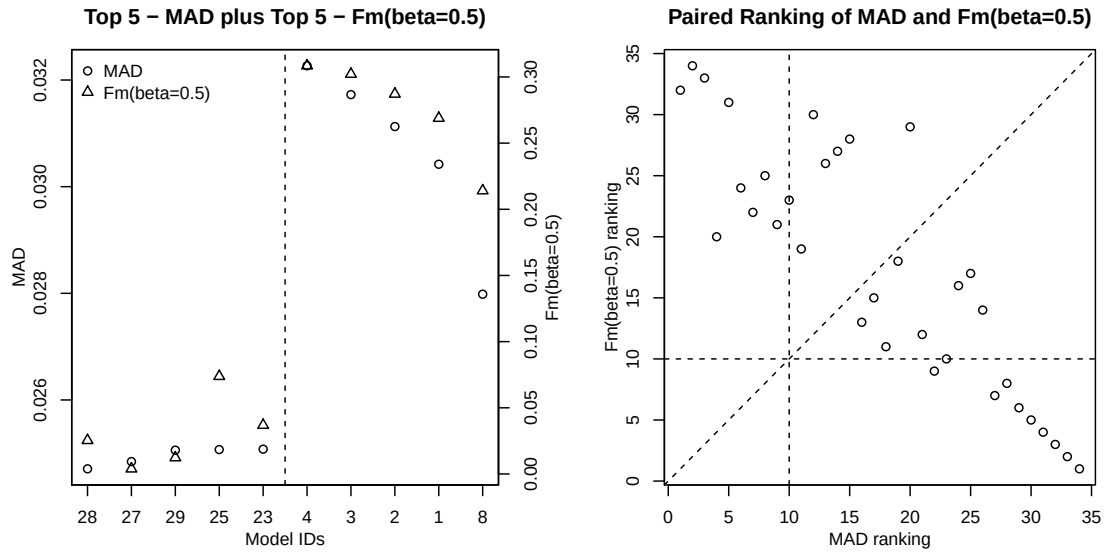


Figure A.6: The results for 5-days returns of Coca-Cola (KO).

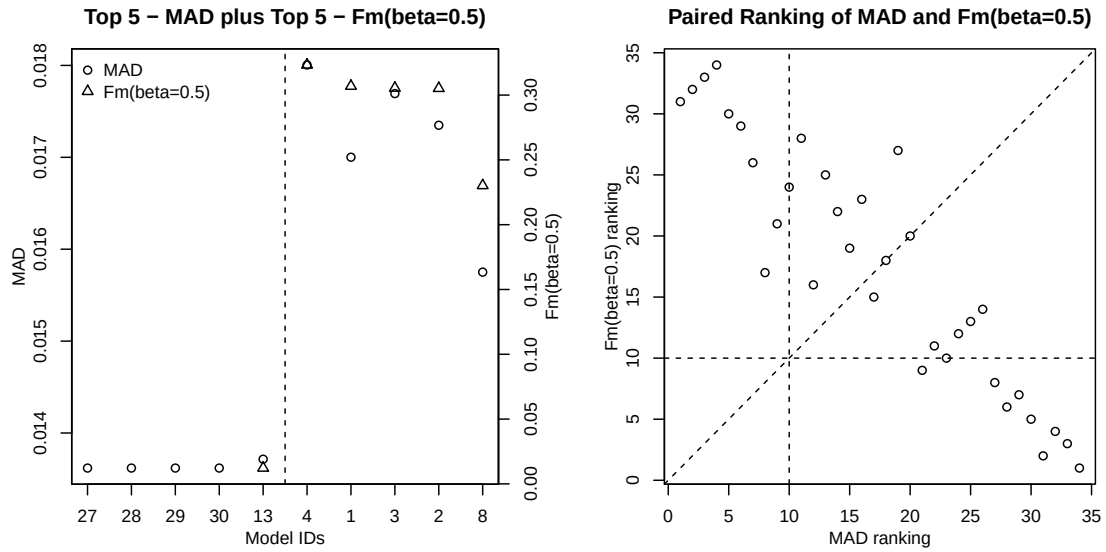


Figure A.7: The results for 1-days returns of Boeing (BA).

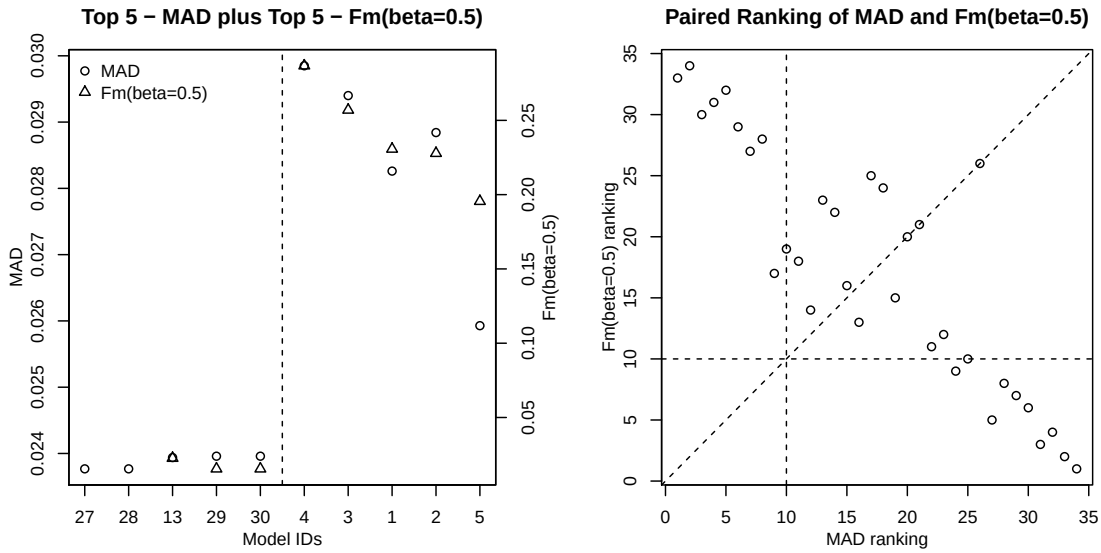


Figure A.8: The results for 3-days returns of Boeing (BA).

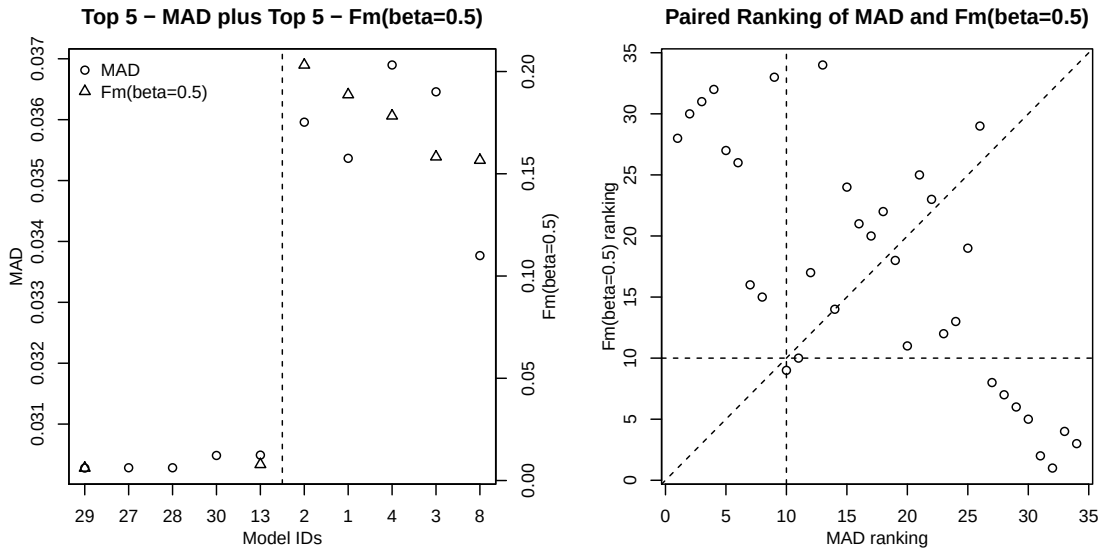


Figure A.9: The results for 5-days returns of Boeing (BA).

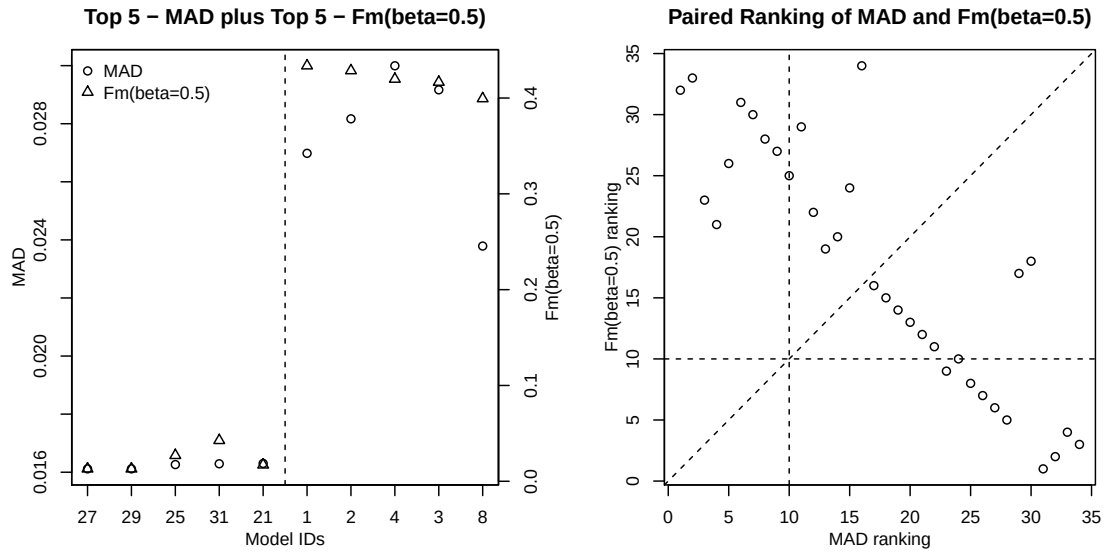


Figure A.10: The results for 1-days returns of General Motors (GM).

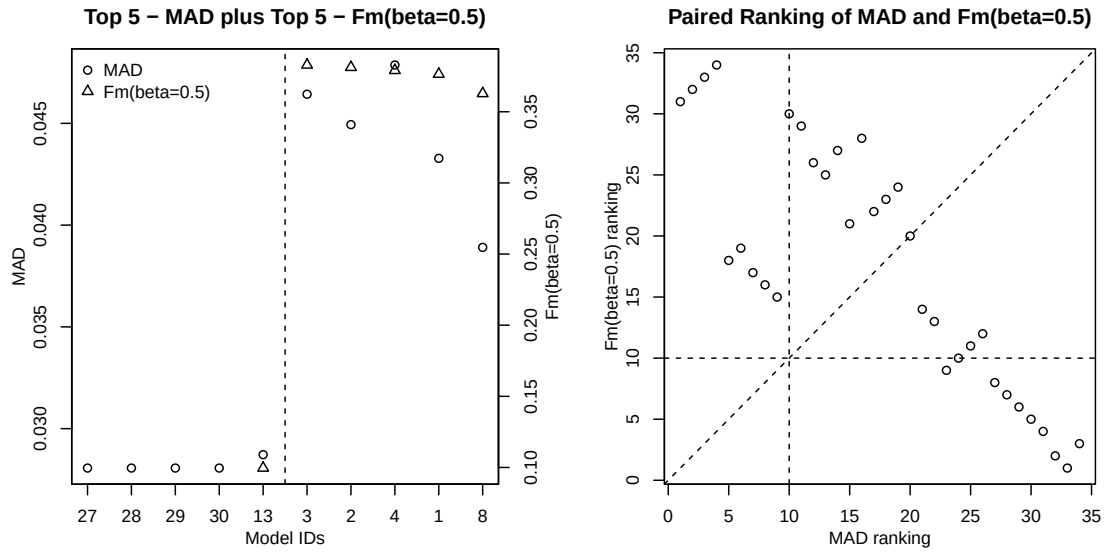


Figure A.11: The results for 3-days returns of General Motors (GM).

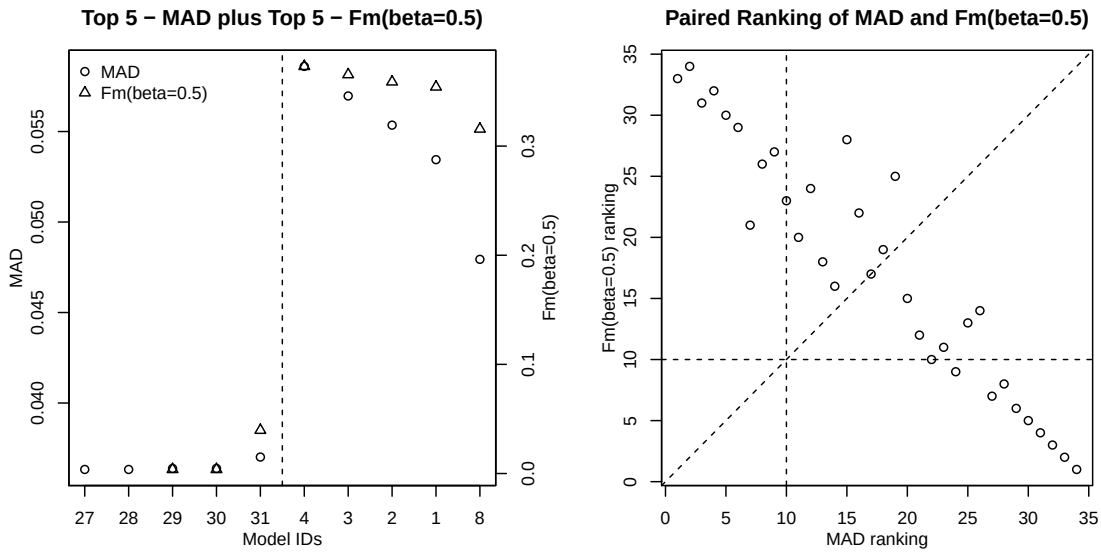


Figure A.12: The results for 5-days returns of General Motors (GM).

- B -

Results of the Experiments with ubaRules

In this appendix, we present the complete set of results obtained from the experiments conducted for the empirical analysis of ubaRules presented in Chapter 6. In Section B.1 we show the utility performance estimates obtained by the systems tested in our experiment. In Section B.2 we present the performance estimates obtained by ubaRules in comparison to cost-sensitive classification techniques also included in the experiments. Finally, in Section B.3 we list the rule set sizes of the models generated by ubaRules and rulefit, the two rule-based modelling tools included in the experiments.

B.1 Utility Estimates

Table B.1: $\text{Fm}_{\beta=0.5}$ performance estimates and average ranks (in parenthesis) within miscellaneous domains data (MISC). The first column reports the average rank obtained by each model across all the datasets.

$\text{Fm}_{\beta=0.5}$ **Performance Estimates (%) and Ranks obtained per Data set**							
Model	**Avg. Rank**	NO2Emissions		ForestFires		Boston	
cart	**7.0**	17.7	(13.0)	36.5	(5.0)	58.3	(3.0)
randomF	**10.3**	19.8	(11.0)	35.9	(6.0)	46.9	(14.0)
svm	**7.7**	10.5	(14.0)	41.3	(4.0)	52.5	(5.0)
mars	**4.3**	20.1	(10.0)	46.6	(2.0)	59.4	(1.0)
rulefit	**5.0**	19.2	(12.0)	46.8	(1.0)	58.7	(2.0)
ubaCart	**2.7**	23.2	(1.0)	42.5	(3.0)	56.4	(4.0)
ubaRbagg100All	**6.8**	22.2	(2.5)	34.6	(9.0)	52.4	(9.0)
ubaRbagg100T0.9	**6.0**	22.2	(4.0)	35.2	(7.0)	52.4	(7.0)
ubaRboost100All	**8.0**	21.9	(6.0)	34.6	(10.0)	52.4	(8.0)
ubaRboost100T0.9	**7.0**	21.9	(7.0)	35.2	(8.0)	52.4	(6.0)
ubaRbagg200All	**11.7**	21.3	(8.0)	34.3	(14.0)	52.4	(13.0)
ubaRbagg200T0.9	**8.5**	22.2	(2.5)	34.4	(12.0)	52.4	(11.0)
ubaRboost200All	**11.3**	21.2	(9.0)	34.3	(13.0)	52.4	(12.0)
ubaRboost200All	**8.7**	22.0	(5.0)	34.6	(11.0)	52.4	(10.0)

Table B.2: $Fm_{\beta=0.5}$ performance estimates and average ranks (in parenthesis) within Harmful Algae Blooms data (HAB) . The first column reports the average rank obtained by each model across all the datasets.

$Fm_{\beta=0.5}$ **Performance Estimates (%) and Ranks obtained per Data set**												
Model	**Avg. Rank**	Cyanobacteria		Chlorophyta		Cryptophyta		Chrysophyta		Diatom		Dinophyta	
cart	**10.7**	0.0	(8.5)	14.4	(5.0)	0.0	(13.5)	4.8	(9.0)	3.3	(14.0)	0.0	(14.0)
randomF	**7.2**	0.0	(8.5)	38.9	(1.0)	14.8	(10.0)	0.0	(12.0)	27.9	(2.0)	16.7	(10.0)
svm	**5.2**	5.3	(1.0)	24.9	(3.0)	0.0	(13.5)	0.0	(12.0)	31.5	(1.0)	29.9	(1.0)
mars	**8.5**	2.9	(2.0)	12.0	(14.0)	33.4	(1.0)	0.0	(12.0)	17.7	(11.0)	7.3	(11.0)
rulefit	**9.8**	0.0	(8.5)	30.0	(2.0)	10.4	(11.0)	0.0	(12.0)	8.5	(13.0)	7.0	(12.0)
ubaCart	**11.8**	0.0	(8.5)	12.9	(13.0)	5.9	(12.0)	0.0	(12.0)	13.2	(12.0)	2.6	(13.0)
ubaRbagg100All	**7.0**	0.0	(8.5)	13.0	(12.0)	21.1	(7.5)	21.0	(1.5)	18.8	(8.0)	24.7	(4.5)
ubaRbagg100T0.9	**6.0**	0.0	(8.5)	14.5	(4.0)	21.1	(7.5)	21.0	(1.5)	19.3	(6.0)	24.6	(8.5)
ubaRboost100All	**6.6**	0.0	(8.5)	13.1	(11.0)	21.2	(2.5)	20.9	(7.5)	18.8	(7.0)	24.7	(3.0)
ubaRboost100T0.9	**6.1**	0.0	(8.5)	14.2	(6.0)	21.2	(2.5)	20.9	(7.5)	19.3	(5.0)	24.6	(7.0)
ubaRbagg200All	**6.8**	0.0	(8.5)	13.5	(8.0)	21.1	(7.5)	21.0	(3.5)	18.1	(9.0)	24.7	(4.5)
ubaRbagg200T0.9	**7.5**	0.0	(8.5)	13.5	(7.0)	21.1	(7.5)	21.0	(3.5)	18.0	(10.0)	24.6	(8.5)
ubaRboost200All	**5.8**	0.0	(8.5)	13.1	(10.0)	21.1	(4.5)	20.9	(5.5)	21.9	(4.0)	24.7	(2.0)
ubaRboost200T0.9	**6.1**	0.0	(8.5)	13.2	(9.0)	21.1	(4.5)	20.9	(5.5)	22.3	(3.0)	24.6	(6.0)

Table B.3: $Fm_{\beta=0.5}$ performance estimates and average ranks (in parenthesis) within Stock Market data (SM) . The first column reports the average rank obtained by each model across all the datasets.

$Fm_{\beta=0.5}$ **Performance Estimates (%) and Ranks obtained per Data set**																									
Model	**Avg. Rank**	IBM.1D.data		IBM.3D.data		IBM.5D.data		KO.1D.data		KO.3D.data		KO.5D.data		BA.1D.data		BA.3D.data		BA.5D.data		GM.1D.data		GM.3D.data		GM.5D.data	
cart	**9.2**	0.0	(8.5)	0.0	(10.5)	0.0	(8.5)	0.0	(12.5)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	0.0	(12.5)	0.0	(8.5)
randomF	**8.5**	0.0	(8.5)	0.0	(10.5)	0.0	(8.5)	4.0	(4.0)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	0.0	(12.5)	0.0	(8.5)
svm	**3.0**	19.8	(1.0)	8.0	(2.0)	15.3	(1.0)	4.0	(8.0)	2.5	(2.0)	2.4	(2.0)	0.0	(7.5)	5.4	(1.0)	0.0	(7.5)	14.1	(1.0)	9.2	(2.0)	16.8	(1.0)
mars	**8.4**	0.0	(8.5)	0.0	(10.5)	0.0	(8.5)	3.9	(9.0)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	1.8	(3.0)	0.0	(12.5)	0.0	(8.5)
rulefit	**9.2**	0.0	(8.5)	0.0	(10.5)	0.0	(8.5)	0.0	(12.5)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	0.0	(12.5)	0.0	(8.5)
ubaCart	**4.0**	9.5	(2.0)	10.9	(1.0)	10.9	(2.0)	0.0	(12.5)	8.7	(1.0)	11.8	(1.0)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	11.6	(2.0)	16.7	(1.0)	7.1	(2.0)
ubaRbagg100All	**8.2**	0.0	(8.5)	0.0	(10.5)	0.0	(8.5)	4.0	(6.0)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	4.0	(7.0)	0.0	(8.5)
ubaRbagg100T0.9	**8.3**	0.0	(8.5)	0.0	(10.5)	0.0	(8.5)	4.0	(6.0)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	3.9	(9.0)	0.0	(8.5)
ubaRboost100All	**7.2**	0.0	(8.5)	1.7	(5.5)	0.0	(8.5)	4.4	(1.5)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	4.1	(5.0)	0.0	(8.5)
ubaRboost100T0.9	**7.6**	0.0	(8.5)	1.7	(5.5)	0.0	(8.5)	4.4	(1.5)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	3.9	(10.0)	0.0	(8.5)
ubaRbagg200All	**8.5**	0.0	(8.5)	0.0	(10.5)	0.0	(8.5)	3.1	(10.0)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	4.0	(7.0)	0.0	(8.5)
ubaRbagg200T0.9	**7.9**	0.0	(8.5)	0.0	(10.5)	0.0	(8.5)	4.0	(6.0)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	4.1	(3.5)	0.0	(8.5)
ubaRboost200All	**8.2**	0.0	(8.5)	1.7	(4.0)	0.0	(8.5)	0.0	(12.5)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	4.0	(7.0)	0.0	(8.5)
ubaRboost200T0.9	**7.0**	0.0	(8.5)	1.7	(3.0)	0.0	(8.5)	4.3	(3.0)	0.0	(8.5)	0.0	(8.5)	0.0	(7.5)	0.0	(8.0)	0.0	(7.5)	0.0	(9.0)	4.1	(3.5)	0.0	(8.5)

B.2 Regression vs Classification

Table B.4: $Fm_{\beta=0.5}$ performance estimates and respective average ranks (in parenthesis) within miscellaneous data domains (MISC) obtained by ubaRules and some classification techniques. The first column reports the average rank obtained by each model across all the datasets.

$Fm_{\beta=0.5}$ **Performance Estimates (%) and Ranks obtained per Data set**							
Model	**Avg. Rank**	NO2Emissions		ForestFires		Boston	
randomF.class	**4.2**	3.6	(9.5)	48.8	(2.0)	70.6	(1.0)
svm.class	**4.2**	3.6	(9.5)	64.9	(1.0)	64.7	(2.0)
ubaRbagg100All	**4.2**	22.2	(1.5)	34.6	(5.0)	52.4	(6.0)
ubaRbagg100T0.9	**3.3**	22.2	(3.0)	35.2	(3.0)	52.4	(4.0)
ubaRboost100All	**5.3**	21.9	(5.0)	34.6	(6.0)	52.4	(5.0)
ubaRboost100T0.9	**4.3**	21.9	(6.0)	35.2	(4.0)	52.4	(3.0)
ubaRbagg200All	**9.0**	21.3	(7.0)	34.3	(10.0)	52.4	(10.0)
ubaRbagg200T0.9	**5.8**	22.2	(1.5)	34.4	(8.0)	52.4	(8.0)
ubaRboost200All	**8.7**	21.2	(8.0)	34.3	(9.0)	52.4	(9.0)
ubaRboost200T0.9	**6.0**	22.0	(4.0)	34.6	(7.0)	52.4	(7.0)

Table B.5: $Fm_{\beta=0.5}$ performance estimates and respective average ranks (in parenthesis) within Harmful Algae Blooms data (HAB) obtained by ubaRules and some classification techniques. The first column reports the average rank obtained by each model across all the datasets.

$Fm_{\beta=0.5}$ **Performance Estimates (%) and Ranks obtained per Data set**													
Model	**Avg. Rank**	Cyanobacteria		Chlorophyta		Cryptophyta		Chrysophyta		Diatom		Dinophyta	
randomF.class	**7.6**	0.0	(5.5)	4.6	(10.0)	0.0	(9.5)	0.0	(9.5)	26.7	(1.0)	0.0	(10.0)
svm.class	**7.1**	0.0	(5.5)	22.6	(1.0)	0.0	(9.5)	0.0	(9.5)	18.6	(8.0)	6.3	(9.0)
ubaRbagg100All	**5.5**	0.0	(5.5)	13.0	(9.0)	21.1	(6.5)	21.0	(1.5)	18.8	(7.0)	24.7	(3.5)
ubaRbagg100T0.9	**4.7**	0.0	(5.5)	14.5	(2.0)	21.1	(6.5)	21.0	(1.5)	19.3	(5.0)	24.6	(7.5)
ubaRboost100All	**5.1**	0.0	(5.5)	13.1	(8.0)	21.2	(1.5)	20.9	(7.5)	18.8	(6.0)	24.7	(2.0)
ubaRboost100T0.9	**4.6**	0.0	(5.5)	14.2	(3.0)	21.2	(1.5)	20.9	(7.5)	19.3	(4.0)	24.6	(6.0)
ubaRbagg200All	**5.5**	0.0	(5.5)	13.5	(5.0)	21.1	(6.5)	21.0	(3.5)	18.1	(9.0)	24.7	(3.5)
ubaRbagg200T0.9	**6.2**	0.0	(5.5)	13.5	(4.0)	21.1	(6.5)	21.0	(3.5)	18.0	(10.0)	24.6	(7.5)
ubaRboost200All	**4.2**	0.0	(5.5)	13.1	(7.0)	21.1	(3.5)	20.9	(5.5)	21.9	(3.0)	24.7	(1.0)
ubaRboost200T0.9	**4.6**	0.0	(5.5)	13.2	(6.0)	21.1	(3.5)	20.9	(5.5)	22.3	(2.0)	24.6	(5.0)

Table B.6: $\text{Fm}_{\beta=0.5}$ performance estimates and respective average ranks (in parenthesis) within Stock Market data (SM) obtained by ubaRules and some classification techniques. The first column reports the average rank obtained by each model across all the datasets.

	$\text{Fm}_{\beta=0.5}$ **Performance Estimates (%) and Ranks obtained per Data set**																							
Model	**Avg. Rank**	IBM.1D.data		IBM.3D.data		IBM.5D.data		KO.1D.data		KO.3D.data		KO.5D.data		BA.1D.data		BA.3D.data		BA.5D.data		GM.1D.data		GM.3D.data		GM.5D.data	
randomF.class	**4.9**	0.0	(5.5)	1.5	(5.0)	0.0	(5.5)	0.0	(9.0)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	6.0	(1.0)	0.0	(5.5)	4.5	(1.0)	2.3	(9.0)	1.9	(1.0)
svm.class	**6.5**	0.0	(5.5)	0.0	(8.0)	0.0	(5.5)	0.0	(9.0)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	0.0	(6.0)	0.0	(5.5)	0.0	(6.0)	0.0	(10.0)	0.0	(6.0)
ubaRbagg100All	**5.8**	0.0	(5.5)	0.0	(8.0)	0.0	(5.5)	4.0	(5.0)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	0.0	(6.0)	0.0	(5.5)	0.0	(6.0)	4.0	(5.0)	0.0	(6.0)
ubaRbagg100T0.9	**5.9**	0.0	(5.5)	0.0	(8.0)	0.0	(5.5)	4.0	(5.0)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	0.0	(6.0)	0.0	(5.5)	0.0	(6.0)	3.9	(7.0)	0.0	(6.0)
ubaRboost100All	**4.9**	0.0	(5.5)	1.7	(3.5)	0.0	(5.5)	4.4	(1.5)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	0.0	(6.0)	0.0	(5.5)	0.0	(6.0)	4.1	(3.0)	0.0	(6.0)
ubaRboost100T0.9	**5.3**	0.0	(5.5)	1.7	(3.5)	0.0	(5.5)	4.4	(1.5)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	0.0	(6.0)	0.0	(5.5)	0.0	(6.0)	3.9	(8.0)	0.0	(6.0)
ubaRbagg200All	**5.9**	0.0	(5.5)	0.0	(8.0)	0.0	(5.5)	3.1	(7.0)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	0.0	(6.0)	0.0	(5.5)	0.0	(6.0)	4.0	(5.0)	0.0	(6.0)
ubaRbagg200T0.9	**5.5**	0.0	(5.5)	0.0	(8.0)	0.0	(5.5)	4.0	(5.0)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	0.0	(6.0)	0.0	(5.5)	0.0	(6.0)	4.1	(1.5)	0.0	(6.0)
ubaRboost200All	**5.6**	0.0	(5.5)	1.7	(2.0)	0.0	(5.5)	0.0	(9.0)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	0.0	(6.0)	0.0	(5.5)	0.0	(6.0)	4.0	(5.0)	0.0	(6.0)
ubaRboost200T0.9	**4.7**	0.0	(5.5)	1.7	(1.0)	0.0	(5.5)	4.3	(3.0)	0.0	(5.5)	0.0	(5.5)	0.0	(5.5)	0.0	(6.0)	0.0	(5.5)	0.0	(6.0)	4.1	(1.5)	0.0	(6.0)

B.3 Rule Set Sizes

Table B.7: Rule set size and respective average ranks (in parenthesis) within miscellaneous domains data (MISC). The first column reports the average rank obtained by each model across all the datasets.

Rule set size Estimates and Ranks obtained per Data set							
Model	Avg. Rank	NO2Emissions		ForestFires		Boston	
rulefit	**8.3**	179.1	(7.0)	579.5	(9.0)	300.6	(9.0)
ubaRbagg100All	**3.5**	164.3	(3.5)	269.8	(3.5)	59.7	(3.5)
ubaRbagg100T0.9	**1.5**	147.9	(1.5)	242.8	(1.5)	53.8	(1.5)
ubaRboost100All	**3.5**	164.3	(3.5)	269.8	(3.5)	59.7	(3.5)
ubaRboost100T0.9	**1.5**	147.9	(1.5)	242.8	(1.5)	53.8	(1.5)
ubaRbagg200All	**7.8**	187.4	(8.5)	410.3	(7.5)	72.0	(7.5)
ubaRbagg200T0.9	**5.5**	168.7	(5.5)	369.3	(5.5)	64.7	(5.5)
ubaRboost200All	**7.8**	187.4	(8.5)	410.3	(7.5)	72.0	(7.5)
ubaRboost200T0.9	**5.5**	168.7	(5.5)	369.3	(5.5)	64.7	(5.5)

Table B.8: Rule set size and respective average ranks (in parenthesis) within Harmful Algae Blooms data (HAB). The first column reports the average rank obtained by each model across all the datasets.

Rule set size Estimates and Ranks obtained per Data set													
Model	Avg. Rank	Cyanobacteria		Chlorophyta		Cryptophyta		Chrysophyta		Diatom		Dinophyta	
rulefit	**9.0**	312.6	(9.0)	274.2	(9.0)	260.1	(9.0)	201.3	(9.0)	298.5	(9.0)	237.2	(9.0)
ubaRbagg100All	**3.5**	159.2	(3.5)	94.5	(3.5)	176.4	(3.5)	173.7	(3.5)	163.0	(3.5)	140.3	(3.5)
ubaRbagg100T0.9	**1.5**	143.8	(1.5)	85.5	(1.5)	159.1	(1.5)	155.5	(1.5)	146.5	(1.5)	126.4	(1.5)
ubaRboost100All	**3.5**	159.2	(3.5)	94.5	(3.5)	176.4	(3.5)	173.7	(3.5)	163.0	(3.5)	140.3	(3.5)
ubaRboost100T0.9	**1.5**	143.8	(1.5)	85.5	(1.5)	159.1	(1.5)	155.5	(1.5)	146.5	(1.5)	126.4	(1.5)
ubaRbagg200All	**7.5**	185.3	(7.5)	126.9	(7.5)	205.3	(7.5)	200.8	(7.5)	184.1	(7.5)	164.6	(7.5)
ubaRbagg200T0.9	**5.5**	166.3	(5.5)	114.6	(5.5)	185.2	(5.5)	179.7	(5.5)	165.3	(5.5)	148.3	(5.5)
ubaRboost200All	**7.5**	185.3	(7.5)	126.9	(7.5)	205.3	(7.5)	200.8	(7.5)	184.1	(7.5)	164.6	(7.5)
ubaRboost200T0.9	**5.5**	166.3	(5.5)	114.6	(5.5)	185.2	(5.5)	179.7	(5.5)	165.3	(5.5)	148.3	(5.5)

Table B.9: Rule set size and respective average ranks (in parenthesis) within Stock Market data (SM). The first column reports the average rank obtained by each model across all the datasets.

Rule set size Estimates and Ranks obtained per Data set																									
Model	Avg. Rank	IBM.1D.data		IBM.3D.data		IBM.5D.data		KO.1D.data		KO.3D.data		KO.5D.data		BA.1D.data		BA.3D.data		BA.5D.data		GM.1D.data		GM.3D.data		GM.3D.data	
rulefit	**2.0**	0.0	(1.0)	0.0	(1.0)	144.0	(1.0)	0.0	(1.0)	196.0	(9.0)	130.0	(5.0)	0.0	(1.0)	0.0	(1.0)	0.0	(1.0)	0.0	(1.0)	0.0	(1.0)	148.0	(1.0)
ubaRbagg100All	**4.3**	19.0	(4.5)	229.0	(4.5)	288.0	(4.5)	57.0	(4.5)	46.0	(3.5)	106.0	(3.5)	164.0	(4.5)	295.0	(4.5)	124.0	(4.5)	62.0	(4.5)	190.0	(4.5)	321.0	(4.5)
ubaRbagg100T0.9	**2.3**	17.0	(2.5)	206.0	(2.5)	259.0	(2.5)	51.0	(2.5)	41.0	(1.5)	95.0	(1.5)	148.0	(2.5)	266.0	(2.5)	112.0	(2.5)	56.0	(2.5)	171.0	(2.5)	289.0	(2.5)
ubaRboost100All	**4.3**	19.0	(4.5)	229.0	(4.5)	288.0	(4.5)	57.0	(4.5)	46.0	(3.5)	106.0	(3.5)	164.0	(4.5)	295.0	(4.5)	124.0	(4.5)	62.0	(4.5)	190.0	(4.5)	321.0	(4.5)
ubaRboost100T0.9	**2.3**	17.0	(2.5)	206.0	(2.5)	259.0	(2.5)	51.0	(2.5)	41.0	(1.5)	95.0	(1.5)	148.0	(2.5)	266.0	(2.5)	112.0	(2.5)	56.0	(2.5)	171.0	(2.5)	289.0	(2.5)
ubaRbagg200All	**8.4**	45.0	(8.5)	269.0	(8.5)	433.0	(8.5)	93.0	(8.5)	60.0	(7.5)	174.0	(8.5)	237.0	(8.5)	383.0	(8.5)	167.0	(8.5)	182.0	(8.5)	329.0	(8.5)	486.0	(8.5)
ubaRbagg200T0.9	**6.4**	40.0	(6.5)	242.0	(6.5)	390.0	(6.5)	84.0	(6.5)	54.0	(5.5)	157.0	(6.5)	213.0	(6.5)	345.0	(6.5)	150.0	(6.5)	164.0	(6.5)	296.0	(6.5)	437.0	(6.5)
ubaRboost200All	**8.4**	45.0	(8.5)	269.0	(8.5)	433.0	(8.5)	93.0	(8.5)	60.0	(7.5)	174.0	(8.5)	237.0	(8.5)	383.0	(8.5)	167.0	(8.5)	182.0	(8.5)	329.0	(8.5)	486.0	(8.5)
ubaRboost200T0.9	**6.4**	40.0	(6.5)	242.0	(6.5)	390.0	(6.5)	84.0	(6.5)	54.0	(5.5)	157.0	(6.5)	213.0	(6.5)	345.0	(6.5)	150.0	(6.5)	164.0	(6.5)	296.0	(6.5)	437.0	(6.5)

References

Abe, N. (2005). Machine learning paradigms for utility-based data mining. In *UBDM'05: Proceedings of the 1st International Workshop on Utility-Based Data Mining*, page 2.

Aldrin, M. and Haff, I. H. (2005). Generalised additive modelling of air pollution, traffic volume and meteorology. *Atmospheric Environment*, 39(11):2145 – 2155. Elsevier Science Inc.

Altidor, W., Khoshgoftaar, T. M., and Napolitano, A. (2009). Wrapper-based feature ranking for software engineering metrics. In *International Conference on Machine Learning and Applications, ICMLA 2009*, pages 241–246. IEEE Computer Society.

Angiulli, F. and Pizzuti, C. (2002). Fast outlier detection in high dimensional space. In Elomaa, T., Mannila, H., and Toivonen, H., editors, *PKDD'02: Proceedings of the 6th European Conference of Principles and Practice of Knowledge Discovery in Databases*, volume 2431 of *LNCS*, pages 15–26. Springer.

Angiulli, F. and Pizzuti, C. (2005). Outlier mining in large high-dimensional data sets. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):203–215. IEEE Computer Society.

Barnett, V. and Lewis, T. (1994). *Outliers in Statistical Data*. Wiley Series in Probability & Statistics.

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

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine Learning*, 1(45):5–32. Springer.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and Regression Trees*. Chapman & Hall/CRC.

Breunig, M. M., Kriegel, H. P., Ng, R., and Sander, J. (1999). Optics-of: Identifying local outliers. In Zytkow, J. M. and Rauch, J., editors, *PKDD'99: Proceedings of 3rd European Conference on Principles of Data Mining and. Knowledge Discovery*, volume 1704 of *LNCS*, pages 262–270. Springer.

Breunig, M. M., Kriegel, H. P., Ng, R., and Sander, J. (2000). Lof: Identifying density-based local outliers. In Chen, W., Naughton, J. F., and Bernstein, P. A., editors, *Proceedings of ACM SIGMOD 2000 International Conference on Management of Data*. ACM Press.

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78:1–3. American Meteorological Society.

Brodersen, K. H., Ong, C. S., Stephan, K. E., and Buhmann, J. M. (2010). The binormal assumption on precision-recall curves. In *20th International Conference on Pattern Recognition*, pages 4263–4266. IEEE.

Chan, P. and Stolfo, S. (1998). Toward scalable learning with non-uniform class and cost distributions: a case study in credit card fraud detection. In Rakesh Agrawal, P. E. S. and Piatetsky-Shapiro, G., editors, *KDD'98: Proceedings of the 4th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 164–168. AAAI Press.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection : A survey. *ACM Computing Surveys*, 41(3):1–58. ACM Press.

Chawla, N. V. (2005). Data mining for imbalanced datasets: An overview. In Maimon, O. and Rokach, L., editors, *The Data Mining and Knowledge Discovery Handbook*, pages 853–867. Springer.

Chawla, N. V., Bowyer, K. W., Hall, O. L., , and Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357. AAAI Press.

Chawla, N. V., Lazarevic, A., Hall, L. O., and Bowyer, K. W. (2003). Smoteboost: improving prediction of the minority class in boosting. In Lavrac, N., Gamberger, D., Blockeel, H., and Todorovski, L., editors, *PKDD'03: Proceedings of 7th European Conference on Principles and Practice of Knowledge Discovery in Databases*, number 2838 in LNCS, pages 107–119. Springer.

Chen, S., Wang, W., and van Zuylen, H. (2010). A comparison of outlier detection algorithms for its data. *Expert Systems with Applications*, 37(2):1169–1178. Pergamon Press, Inc.

Cheng, T. and Li, Z. (2006). A multiscale approach for spatio-temporal outlier detection. *Transactions in GIS*, 10(2):253 – 263. Wiley.

Cherkassky, V. and Ma, Y. (2004). Practical selection of svm parameters and noise estimation for svm regression. *Neural Networks*, 17:113–126.

Christoffersen, P. F. and Diebold, F. X. (1996). Further results on forecasting and model selection under asymmetric loss. *Journal of Applied Econometrics*, 11:561–571. Wiley.

Clark, P. and Niblett, T. (1989). The cn2 induction algorithm. *Machine Learning*, 3:261–283.

Clémençon, S. and Vayatis, N. (2009). Nonparametric estimation of the precision-recall curve. In Danyluk, A. P., Bottou, L., and Littman, M. L., editors, *Proceedings of the 26th Annual*

International Conference on Machine Learning, volume 382 of *ACM International Conference Proceeding Series*, pages 24–31. ACM.

Cleveland, W. (1993). *Visualizing Data*. Hobart Press.

Cohen, W. W. (1995). Fast Effective Rule Induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123.

Cohen, W. W. and Singer, Y. (1999). A simple, fast, and effective rule learner. In *In Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 335–342. AAAI Press.

Coles, S. (2001). *An Introduction to Statistical Modeling of Extreme Values*. Springer, 1 edition.

Crone, S., Lessmann, S., and Stahlbock, R. (2005). Utility based data mining for time series analysis - cost-sensitive learning for neural networks. In *UBDM'05: Proceedings of the 1st International Workshop on Utility-Based Data Mining*, pages 59–68.

Daskalaki, S., Kopanas, I., and Avouris, N. M. (2006). Evaluation of classifiers for an uneven class distribution problem. *Applied Artificial Intelligence*, 20(5):381–417. Taylor & Francis Group.

Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In Cohen, W. W. and Moore, A., editors, *ICML'06: Proceedings of the 23rd International Conference on Machine Learning*, volume 148 of *ACM ICPS*, pages 233–240. ACM.

Degen, M., Embrechts, P., and Lambrigger, D. D. (2007). The quantitative modeling of operational risk: between g-and-h and evt. *ASTIN Bulletin: The Journal of the International Actuarial Association*, 38:1–32. Peeters Publishers.

Dembczynski, K., Kotłowski, W., and Słowinski, R. (2010). Ender: a statistical framework for boosting decision rules. *Data Mining and Knowledge Discovery*, 21:52–90.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal Machine Learning Research*, 7:1–30. MIT Press.

Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D., , and Weingessel, A. (2010). *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*. R package version 1.5-24.

Domingos, P. (1999). Metacost: A general method for making classifiers cost-sensitive. In Chaudhuri, S., Madigan, D., and Fayyad, U., editors, *KDD'99: Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining*, pages 155–164. ACM Press.

Dougherty, R. L., Edelman, A., and Hyman, J. M. (1989). Nonnegativity-, monotonicity-, or convexity-preserving cubic and quintic Hermite interpolation. *Mathematics of Computation*, 52(186):471–494.

Drucker, H. (1997). Improving regressors using boosting techniques. In Fisher, D. H., editor, *Fourteenth International Conference on Machine Learning*, pages 107–115. Morgan Kaufmann.

Drummond, C. and Holte, R. C. (2000). Explicitly representing expected cost: an alternative to roc representation. In Simoff, S. J. and Zaïane, O. R., editors, *KDD'00: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 198–207. ACM Press.

Drummond, C. and Holte, R. C. (2004). What roc curves can't do (and cost curves can). In Hernández-Orallo, J., Ferri, C., Lachiche, N., and Flach, P. A., editors, *ROCAI'04: 1st International Workshop on ROC Analysis in Artificial Intelligence*, pages 19–26.

Drummond, C. and Holte, R. C. (2006). Cost curves: An improved method for visualizing classifier performance. *Machine Learning*, 65(1):95–130. Springer.

Egan, J. (1975). *Signal detection theory and ROC analysis*. New York: Academic Press.

Elkan, C. (2001). The foundations of cost-sensitive learning. In Nebel, B., editor, *IJCAI'01: Proceedings of 17th International Joint Conference of Artificial Intelligence*, volume 1, pages 973–978. Morgan Kaufmann Publishers Inc.

Embrechts, P., Mikosch, T., and Klüppelberg, C. (1997). *Modelling extremal events: for insurance and finance*. Springer.

Embrechts, P., Resnick, S., and Samorodnitsky, G. (1998). Living on the edge. *Journal of Risk*, 11:96–100. Incisive Media.

Estabrooks, A., Jo, T., and Japkowicz, N. (2004). A multiple resampling method for learning from imbalances data sets. *Computational Intelligence*, 20(1):18–36. Wiley InterScience.

Fan, W., Stolfo, S., Zhang, J., and Chan, P. K. (1999). Adacost: Misclassification cost-sensitive boosting. In Bratko, I. and Dzeroski, S., editors, *ICML'99: Proceedings of 16th International Conference on Machine Learning*, pages 97–105. Morgan Kaufmann Publishers Inc.

Faraway, J. (2008). *faraway: Functions and datasets for books by Julian Faraway.*

Fawcett, T. (2004). Roc graphs: Notes and practical considerations for researchers. Technical report, HP Laboratories.

Fawcett, T. (2006a). An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874. Elsevier Science Inc.

Fawcett, T. (2006b). Roc graphs with instance-varying costs. *Pattern Recognition Letters*, 27(8):882–891. Elsevier Science Inc.

Fawcett, T. and Niculescu-Mizil, A. (2007). Pav and the roc convex hull. *Machine Learning*, 68(1):97–106. Springer.

Ferri, C. and Flach, P. (2002). Learning decision trees using the area under the roc curve. In Sammut, C. and Hoffmann, A. G., editors, *ICML'02: Proceedings of the 19th International Conference on Machine Learning*, pages 139–146. Morgan Kaufmann Publishers Inc.

Flach, P. (2007). Tutorial on roc analysis for ranking and probability estimation. Presented at UAI'07: 23rd Conference on Uncertainty in Artificial Intelligence.

Flach, P. and Lavrac, N. (2003). *Intelligent Data Analysis: An Introduction*, chapter Rule Induction, pages 229–267. Springer-Verlag.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139. Elsevier.

Friedman, C. and Sandow, S. (2010). *Utility-based Learning from Data*. 978-1-58488-622-8. Chapman & Hall/CRC Machine Learning & Pattern Recognition.

Friedman, J. (1991). Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–141. Institute of Mathematical Statistics.

Friedman, J. H. and Popescu, B. E. (2008). Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2(3):916–954. Institute of Mathematical Statistics.

Fritsch, F. N. and Carlson, R. E. (1980). Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17:238–246.

Ghoting, A., Parthasarathy, S., and Otey, M. E. (2008). Fast mining of distance-based outliers in high-dimensional datasets. *Data Mining and Knowledge Discovery*, 16(3):349–364. Springer.

Goadrich, M., Oliphant, L., and Shavlik, J. W. (2006). Gleaner: Creating ensembles of first-order clauses to improve recall-precision curves. *Machine Learning*, 64(1-3):231–261. Springer.

Grubbs, F. (1969). Procedures for detecting outlying observations in samples. *Technometrics*, 11:1–21. American Statistical Association.

Guerraggio, A. and Molho, E. (2004). The origins of quasi-concavity: a development between mathematics and economics. *Historia Mathematica*, 31(1):62 – 75.

Hand, D. J. (2009). Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine Learning*, 77(1):103–123.

Hardle, W. (1990). *Applied Nonparametric Regression*. Cambridge University Press.

Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized additive models*. London: Chapman & Hall.

Hawkins, D. M. (1980). *Identification of Outliers*. Chapman and Hall.

Hendrix, E. M. and Toth, B. G. (2010). *Introduction to Nonlinear and Global Optimization*, volume 37 of *Springer Optimization and Its Applications*. Springer-Verlag, 1st edition.

Hodge, V. J. and Austin, J. (2004). A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22:85–126. Springer.

Holmes, G., Hall, M., and Frank, E. (1999). Generating rule sets from model trees. In *Australian Joint Conference on Artificial Intelligence*, volume 1747 of *Lecture Notes in Computer Science*, pages 1–12. Springer.

Indurkhya, N. and Weiss, S. M. (2001). Solving regression problems with rule-based ensemble classifiers. In *KDD '01: Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 287–292. ACM Press.

Jo, T. and Japkowicz, N. (2004). Class imbalances versus small disjuncts. *SIGKDD Explorations Newsletter*, 6(1):40–49. ACM Press.

Joshi, M. V., Agarwal, R. C., and Kumar, V. (2002). Predicting rare classes: Comparing two-phase rule induction to cost-sensitive boosting. In Elomaa, T., Mannila, H., and Toivonen, H., editors, *PKDD'02: Proceedings of the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 2431 of *LNCS*, pages 237–249. Springer.

Joshi, M. V., Kumar, V., and Agarwal, R. (2001). Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In Cercone, N., Lin, T. Y., and Wu, X., editors, *ICDM'01: Proceedings of 1st IEEE International Conference on Data Mining*, pages 257–264. IEEE Computer Society.

Karalič, A. and Bratko, I. (1997). First order regression. *Machine Learning*, 26:147–176.

Kent, A., Berry, M., Leuhrs, F., and Perry, J. (1955). Machine literature searching viii. operational criteria for designing information retrieval systems. In *American Documentation*, volume 6 of *2*, pages 93–101. Wiley Periodicals, Inc.

Knorr, E. M. and Ng, R. T. (1998). Algorithms for mining distance-based outliers in large datasets. In *VLDB'98: Proceedings of 24th International Conference on Very Large Data Bases*, pages 392–403. Morgan Kaufmann, San Francisco, CA.

Kotsiantis, S., Kanellopoulos, D., and Pintelas, P. (2006). Handling imbalanced datasets: a review. *GESTS International Transactions on Computer Science and Engineering*, 30(1):25–36. GESTS Society.

Krzanowski, W. J. and Hand, D. J. (2009). *ROC Curves for Continuous Data*. Chapman & Hall/CRC, 1st edition.

Kubat, M. and Matwin, S. (1997). Addressing the curse of imbalanced training sets: one-sided selection. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 179–186.

Laurikkala, J., Juhola, M., and Kentala, E. (2000). Informal identification of outliers in medical data. In *IDAMAP'00: 5th International Workshop on Intelligent Data Analysis in Medicine and Pharmacology*, pages 20–24.

Li, X. and Han, J. (2007). Mining approximate top-k subspace anomalies in multi-dimensional time-series data. In Koch, C., Gehrke, J., Garofalakis, M. N., Srivastava, D., Aberer, K., Deshpande, A., Florescu, D., Chan, C. Y., Ganti, V., Kanne, C.-C., Klas, W., and Neuhold, E. J., editors, *VLDB '07: Proceedings of the 33rd International Conference on Very Large Data Bases*, pages 447–458. ACM.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News: The Newsletter of the R project*, 2(3):18–22. R Foundation for Statistical Computing.

Ling, C. X. and Sheng, V. S. (2010). Cost-sensitive learning and the class imbalance problem. Encyclopedia of Machine Learning. To appear.

Ling, C. X., Yang, Q., Wang, J., and Zhang, S. (2004). Decision trees with minimal costs. In Brodley, C. E., editor, *ICML '04: Proceedings of the 21st international conference on Machine learning*, volume 69 of *ACM ICPS*, pages 69–81. ACM.

Liu, X.-Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):63–77. IEEE Computer Society.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Metz, C. E. (1978). Basic principles of roc analysis. *Seminars in Nuclear Medicine*, 8(4):283–298. Elsevier Inc.

Milborrow, S., Hastie, T., and Tibshirani., R. (2010). *earth: Multivariate Adaptive Regression Spline Models*. R package version 2.4-5.

Muthukrishnan, S., Shah, R., and Vitter, J. S. (2004). Mining deviants in time series data streams. In *SSDBM' 04: 16th International Conference on Scientific and Statistical Database Management*, pages 41–50. IEEE Computer Society.

Neumann, J. V. and Morgenstern, O. (1944). *Theory of Games and Economic Behavior*. Princeton University Press.

Papadimitriou, S., Kitagawa, H., Faloutsos, C., and Gibbons, P. B. (2003). Loci: Fast outlier detection using the local correlation integral. In Dayal, U., Ramamritham, K., and Vijayaraman, T. M., editors, *ICDE'03: Proceedings of 19th International Conference on Data Engineering*, pages 315–326. IEEE Computer Society.

Petrie, A. and Sabin, C. (2005). *Medical Statistics at a Glance*. Blackwell Publishing, 2nd edition.

Prati, R. C., Batista, G. E. A. P. A., and Monard, M. C. (2004). Learning with class skews and small disjuncts. In Bazzan, A. L. C. and Labidi, S., editors, *SBIA'04: Advances in Artificial Intelligence - 17th Brazilian Symposium on Artificial Intelligence*, volume 3171 of *LNCS*, pages 296–306. Springer.

Prati, R. C. and Flach, P. A. (2005). Roccer: An algorithm for rule learning based on roc analysis. In Kaelbling, L. P. and Saffiotti, A., editors, *IJCAI'05: Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 823–828. Professional Book Center.

Provost, F. and Fawcett, T. (1997). Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *KDD'97: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, pages 43–48. AAAI Press.

Provost, F. and Fawcett, T. (2001). Robust classification for imprecise environments. *Machine Learning*, 42(3):203 – 231. Springer.

Provost, F. J., Fawcett, T., and Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms. In *ICML'98: Proceedings of the 15th International Conference on Machine Learning*, pages 445–453. Morgan Kaufmann Publishers Inc.

Quinlan, J. R. (1992). Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, pages 343–348.

Quinlan, J. R. (1993). Combining instance-based and model-based learning. In *10th International Conference of Machine Learning*, pages 236–243. Morgan Kaufmann Prublishers.

R Development Core Team (2010). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Ramaswamy, S., Rastoji, R., and Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. In *Proceedings of ACM International Conference on Management of Data (SIGMOD 2000)*, pages 427–438.

Ren, D., Wang, B., and Perrizo, W. (2004). Rdf: A density-based outlier detection method using vertical data representation. In *ICDM '04: Proceedings of the 4th IEEE International Conference on Data Mining*, pages 503–506. IEEE Computer Society.

Ribeiro, R. P. and Torgo, L. (2006). Rule-based prediction of rare extreme values. In et al, T. L., editor, *DS'06: Proceeding of the 9th International Conference on Discovery Science*, volume 4265 of *LNCS*, pages 219–230. Springer. Carl Smith Best Student Paper Award sponsored by Yahoo! Research.

Ribeiro, R. P. and Torgo, L. (2007). An effective evaluation metric for forecasting microalgae blooms. Technical report, LIAAD INESC PORTO LA.

Ribeiro, R. P. and Torgo, L. (2008a). A comparative study on predicting algae blooms in douro river, portugal. *Ecological Modelling*, 212(1-2):86–91. Selected Papers from the 5th European Conference on Ecological Modelling. Elsevier.

Ribeiro, R. P. and Torgo, L. (2008b). Utility-based performance measures for regression. In et al, K. W., editor, *The 3rd Workshop on Evaluation Methods for Machine Learning.* held in the context of the 25th International Conference of Machine Learning (ICML ' 08).

Rijsbergen, C. V. (1979). *Information Retrieval.* Dept. of Computer Science, University of Glasgow, 2nd edition.

Rosset, S., Perlich, C., and Zadrozny, B. (2007). Ranking-based evaluation of regression models. *Knowledge and Information Systems*, 12(3):331–353.

Rousseeuw (1985). *Multivariate estimation with high breakdown point*, volume B. Mathematical Statistics and Applications, Vol. B. Reidel Publishing Company.

Rousseeuw, P., Croux, C., Todorov, V., Ruckstuhl, A., Salibian-Barrera, M., Verbeke, T., and Maechler, M. (2008). *robustbase: Basic Robust Statistics.*

Rousseeuw, P. J. and Leroy, A. M. (1987). *Robust regression and outlier detection.* John Wiley & Sons, Inc.

Schapire, R. E. (1999). A brief introduction to boosting. In Dean, T., editor, *IJCAI'99: Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 1401–1406. Morgan Kaufmann Publishers Inc. Invited Speaker.

Shekhar, S., Lu, C. T., and Zhang, P. (2003). A unified approach to spatial outliers detection. Technical report, Department of Computer Science and Engineering, University of Minnesota.

Sheng, B., Li, Q., Mao, W., and Jin, W. (2007). Outlier detection in sensor networks. In Knightly, E. W., Chiasserini, C.-F., and Lin, X., editors, *MobiHoc '07: Proceedings of the 8th ACM international symposium on Mobile Ad Hoc Networking and Computing*, pages 219–228. ACM.

Sheng, V. S. and Ling, C. X. (2006). Thresholding for making classifiers cost-sensitive. In *AAAI'06: Proceedings of the 21st National Conference on Artificial Intelligence and the 18th Innovative Applications of Artificial Intelligence Conference.* AAAI Press.

Shrestha, D. L. and Solomatine, D. P. (2004). Adaboost.rt: a boosting algorithm for regression problems. In *IJCNN'04: Proceedings of International Joint Conference on Neural Networks*, volume 2. IEEE Computer Society.

Sing, T., Sander, O., Beerenwinkel, N., and Lengauer, T. (2009). *ROCR: Visualizing the performance of scoring classifiers.* R package version 1.0-4.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 36(1):111–147. Wiley.

Subramaniam, S., Palpanas, T., Papadopoulos, D., Kalogeraki, V., and Gunopulos, D. (2006). Online outlier detection in sensor data using non-parametric models. In Dayal, U., Whang, K.-Y., Lomet, D. B., Alonso, G., Lohman, G. M., Kersten, M. L., Cha, S. K., and Kim, Y.-K., editors, *VLDB '06: Proceedings of the 32nd International Conference on Very Large Data Bases*, pages 187–198. ACM.

Sun, P., Chawla, S., and Arunasalam, B. (2006). Mining for outliers in sequential databases. In Ghosh, J., Lambert, D., Skillicorn, D. B., and Srivastava, J., editors, *SDM'06: Proceedings of the 6th SIAM International Conference on Data Mining*. SIAM.

Therneau, T. M., Atkinson, B., and Ripley, B. (2008). *rpart: Recursive Partitioning.* R package version 3.1-39.

Torgo, L. (1995). Data fitting with rule-based regression. In *In Proceedings of the 2nd international workshop on Artificial Intelligence Techniques (AIT'95.*

Torgo, L. (1999). *Inductive Learning of Tree-based Regression Models.* PhD thesis, Department of Computer Science, Faculty of Sciences, University of Porto.

Torgo, L. (2005). Regression error characteristic surfaces. In Grossman, R., Bayardo, R., and Bennett, K., editors, *KDD'05: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 697–702. ACM Press.

Torgo, L. (2007). Resource-bounded fraud detection. In Neves, J., Santos, M. F., and Machado, J., editors, *EPIA' 2007 Workshops: 13th Portuguese Conference on Aritficial Intelligence - Progress in Artificial Intelligence*, volume 4874 of *LNCS*, pages 449–460. Springer.

Torgo, L. (2010). *Data Mining with R, learning with case studies.* Chapman and Hall/CRC.

Torgo, L. and Gama, J. (1997). Regression using classification algorithms. *Intelligent Data Analysis*, 1(1-4):275–292. IOS Press.

Torgo, L. and Ribeiro, R. P. (2007). Utility-based regression. In et al, K. J. N., editor, *PKDD'07: Proceedings of 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 4701 of *LNCS*, pages 597–604. Springer.

Torgo, L. and Ribeiro, R. P. (2009). Precision and recall in regression. In Gama, J., Costa, V. S., Jorge, A. M., and Brazdil, P., editors, *DS'09: 12th International Conference on Discovery Science*, volume 5808 of *LNCS*, pages 332–346. Springer.

Tukey, J. W. (1977). *Exploratory Data Analysis*. Addison-Wesley.

Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer, New York, fourth edition. ISBN 0-387-95457-0.

Weiss, G. and Provost, F. (2001). The effect of class distribution on classifier learning: an empirical study. Technical Report Technical Report ML-TR-44, Department of Computer Science, Rutgers University.

Weiss, G., Saar-Tsechansky, M., and Zadrozny, B., editors (2005). *UBDM'05: Proceedings of the 1st International Workshop on Utility-Based Data Mining*. Held in the context of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD' 05), ACM Press.

Weiss, G., Zadrozny, B., and Saar-Tsechansky, M. (2008). Guest editorial: special issue on utility-based data mining. *Data Mining and Knowledge Discovery*, 17(2):129–135. Springer.

Weiss, G. M. (2004). Mining with rarity: a unifying framework. *SIGKDD Explorations Newsletter*, 6(1):7–19. ACM Press.

Weiss, S. M. and Indurkhya, N. (1995). Rule-based machine learning methods for functional prediction. *Journal of Artificial Intelligence Research*, 3:383–403. AAAI Press.

WHO (2006). Who air quality guidelines for particulate matter, ozone, nitrogen dioxide and sulfur dioxide - global update 2005. summary of risk assessment. Technical Report WHO/SDE/PHE/OEH/06.02, World Health Organization (WHO), Geneva, Switzerland.

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

Yang, C., Duraiswami, R., Gumerov, N. A., and Davis, L. (2003). Improved fast gauss transform and efficient kernel density estimation. In *ICCV'03: Proceedings of 9th IEEE International Conference on Computer Vision*, volume 1, pages 664–671. IEEE Computer Society.

Yuanhong, D., Hongchang, C., and Tao, P. (2009). Cost-sensitive support vector machine based on weighted attribute. In *IFITA'09: International Forum on Information Technology and Applications*, volume 1, pages 690–692. IEEE Computer Society.

Zadrozny, B. (2003). *Policy mining: Learning decision policies from fixed sets of data*. PhD thesis, University of California, San Diego.

Zadrozny, B. (2005). One-benefit learning: cost-sensitive learning with restricted cost information. In Weiss, G., Saar-Tsechansky, M., and Zadrozny, B., editors, *UBDM'05: Proceedings of the 1st International Workshop on Utility-Based Data Mining*, pages 53–58. ACM Press.

Zadrozny, B. and Elkan, C. (2001). Learning and making decisions when costs and probabilities are both unknown. In *KDD'01: Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 204–213. ACM Press.

Zadrozny, B. and Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *KDD'02: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 694–699. ACM Press.

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

Zadrozny, B., Weiss, G., and Saar-Tsechansky, M., editors (2006). *UBDM'06: Proceedings of the 2nd International Workshop on Utility-Based Data Mining*. Held in the context of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD' 06).

Zhang, T., Ramakhrishnan, R., and Livny, M. (1996). Birch: An efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada*, pages 103–114. ACM Press.

Zhang, Y., Meratnia, N., and Havinga, P. (2007). A taxonomy framework for unsupervised outlier detection techniques for multi-type data sets. Technical Report TR-CTIT-07-79, Centre for Telematics and Information Technology, University of Twente.