U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

MSc

MSc

2.º
CICLO

FCUP
2014

U.PORTO

Re-sampling Approaches for Regression Tasks
under Imbalanced Domains

Paula Branco

FC

U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Re-sampling Approaches for Regression Tasks under Imbalanced Domains

Paula Branco

Dissertação de Mestrado apresentada à
Faculdade de Ciências da Universidade do Porto em
Ciência de Computadores

2014

MSc

Re-sampling Approaches for Regression Tasks under Imbalanced Domains

Paula Branco

Mestrado em Ciências de Computadores

Ciências de Computadores

2014

Orientador

Prof. Dr. Luís Torgo

Coorientador

Prof. Dr. Rita Ribeiro

U.PORTO

FC FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, ______/______/_________

MSc

To Nicolau and Leonardo

Acknowledgments

I would like to thank Professor Luís Torgo and Professor Rita Ribeiro for having guided me throughout this thesis. I am sincerely grateful to them for sharing their knowledge and expertise and for the patience and support they have awarded me along this path.

A very special thanks to my family for the encouragement and strength.

The work presented in this thesis was partially funded by LIAAD-INESC Porto through the research grant 948/BI_B2A/14.

Paula Branco

Porto, 2014

Abstract

Many real world domains, such as meteorological and financial, involve obtaining predictive models that should be particularly accurate in a specific sub-range of the domain of the target variable. Frequently, these values are poorly represented in the available data set. In this case, we face a challenge usually known as the problem of imbalanced domains.

The existence of few examples that match the user specific preferences creates important problems at different levels. One of these levels is related with the unsuitability of the existing performance assessment metrics. Another level is the need for approaches that are able to force the algorithms to focus on these rare situations. Both aspects are studied in this thesis.

Considering adequate metrics for this problem type is essential. We start by reviewing the existing performance assessment metrics for imbalanced domains and propose a new formulation specifically for regression tasks, which we then use in the experimental evaluation of different methods for handling these problems.

We then address the problem of regression tasks under imbalanced data distribution using re-sampling methods. An extensive survey of the existing approaches both in classification and regression is presented. Among all the existing types of techniques, re-sampling methods are the most studied for classification tasks. These methods are extremely versatile. In effect, re-sampling approaches simply manipulate the given training set changing the examples distribution. This way, they allow the use of any standard learning system. Still, no effort has been made in this field for regression tasks. In this thesis, we propose three new re-sampling methods to address the problem of imbalanced data distribution for regression tasks.

We have carried out an extensive experimental evaluation of the proposed methods on 18 data sets using a large set of learning systems. Results provide clear evidence of the advantages of using the proposed re-sampling approaches for this type of problems.

Resumo

Muitos domínios reais, como meteorológicos e financeiros, envolvem a obtenção de modelos de previsão que devem ser particularmente precisos num sub-intervalo específico do domínio da variável objetivo. No entanto, muitas vezes, esses valores estão pouco representados no conjunto disponível de dados. Neste caso, estamos perante um desafio que é geralmente conhecido como o problema dos domínios desbalanceados.

A existência de poucos exemplos que satisfaçam as preferências específicas do utilizador gera problemas importantes a diferentes níveis. Um destes níveis está relacionado com a desadequação das métricas de avaliação de desempenho existentes. Noutro patamar encontra-se a necessidade de desenvolver abordagens que sejam capazes de forçar os algoritmos a concentrarem-se nestas situações raras. Ambos os aspetos são estudados nesta tese.

Considerar métricas adequadas a este tipo de problema é essencial. Começamos por rever as métricas de avaliação de desempenho existentes para domínios desbalanceados e propomos uma nova formulação especificamente para tarefas de regressão que depois utilizamos na avaliação experimental de diferentes métodos para lidar com estes problemas.

De seguida, abordamos o problema de tarefas de regressão sob uma distribuição desbalanceada dos dados usando métodos de re-amostragem. É apresentado um levantamento extensivo das abordagens existentes em classificação e regressão. De entre todos os tipos de técnicas existentes, os métodos de re-amostragem são os mais estudados para tarefas de classificação. Estes métodos são extremamente versáteis. Com efeito, as abordagens de re-amostragem simplesmente manipulam o conjunto de treino dado alterando a distribuição dos exemplos. Desta forma, permitem o uso de qualquer sistema de aprendizagem standard. Ainda assim, nenhum esforço foi feito nesse campo para as tarefas de regressão. Nesta tese, propomos três novos métodos de re-amostragem para resolver o problema da distribuição desbalanceada dos dados em tarefas de regressão.

Realizamos uma extensa avaliação experimental dos métodos propostos em 18 conjuntos de dados utilizando um grande conjunto de sistemas de aprendizagem. Os resultados fornecem uma evidência clara das vantagens da utilização das abordagens de re-amostragem propostas neste tipo de problema.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Context and Problem Definition

Predictive modelling tasks provide the context for the problem of imbalanced domains. These tasks have the goal of constructing a model of an unknown function to accurately forecast the values of a target variable. When the user is interested in a specific subrange of the target variable values, and there are few examples for that particular subset, we face a challenge known as the problem of imbalanced domains. These tasks raise two main problems that must be addressed together: i) the standard evaluation metrics are no longer adequate, and ii) new approaches are needed to force the learning algorithms to focus on the more important and least represented cases.

The problem of imbalanced domains was extensively studied for classifications tasks where the target variable is nominal. Several performance assessment metrics were provided and many types of approaches were proposed. Re-sampling methods are among the more popular approaches for coping with the class imbalance problem. Yet, little attention has been given to regression tasks where the distribution of the numeric target variable is imbalanced.

In this thesis, we address the problem of imbalanced data distributions for regression tasks through re-sampling methods.

1.2 Motivation and Main Contributions

The problem of imbalanced data distributions is extremely relevant for several real world applications, such as finance, ecology, medicine, telecommunications, web, meteorology, etc. and, therefore, has been getting more attention in recent years. Existing work at the modeling level is focused on classification tasks and is already formed by several categories of approaches. However, no attention has been given to regression tasks under imbalanced data distributions.

Among the different existing approaches to handle distribution imbalance on the target variable, re-sampling methods have the key advantage of being independent of the modelling technique and thus generally applicable. This thesis studies the application of these strategies to regression tasks.

The main contributions of this work are: i) to highlight the importance of considering adequate metrics for this problem type; ii) present the state of the art on performance assessment metrics and approaches for imbalanced data sets; iii) to provide an extensive survey of the existing approaches to tackle the problem of imbalanced domains for classification and regressiontasks; and iv) propose and perform an experimental analysis of three re-sampling methods for addressing regression problems under imbalanced data distributions.

1.3 Organization of the Thesis

The thesis is structured in six chapters whose contents are briefly described below. The present chapter briefly describes the problem addressed in the thesis, and also the motivation and main contributions. In Chapter 2 the problem of imbalanced domains is presented along with a discussion on related problems. Chapter 3 describes the state of the art of performance assessment metrics for both classification and regression tasks under imbalanced domains. In Chapter 4 we continue with the study of the state of the art of approaches to deal with this problem. We present a survey on this topic covering different classes of strategies. Chapter 5 describes our proposal of re-sampling strategies for regression tasks under imbalanced domains. We present three algorithms to tackle this problem and evaluate their performance. Finally, Chapter 6 concludes the thesis and outlines some possible future work.

Chapter 2

The Problem of Imbalanced Domains

2.1 Problem Definition

The problem of imbalanced domains occurs in the context of predictive tasks. Predictive modelling tasks are data analysis tasks with the goal of building a model that provides a good approximation of an unknown function $Y = f(X_1, X_2, \cdots, X_p)$, which maps the values of a set of p predictor variables into the value of a target variable. The model is obtained based on a training data set $D = \{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{n}$. Depending on the type of variable Y, we face either a classification task (nominal Y) or a regression task (numeric Y). For constructing the model an optimization process is used that tries to find the "optimal" model parameters according to some predefined criterion. The most frequently used criteria are the error rate for classification and the mean squared error for regression.

For many real world applications there is a specific subset of the range of values of the target variable Y which is more important, i.e., it is more relevant that the models are particularly accurate in a given sub-range of the target variable domain. Examples include diagnostic of rare diseases or forecasting rare extreme returns on financial markets, among many other.

Moreover, this higher relevance of some subset of the values is often associated with rarity of these values. In these cases we face what is usually known as a problem of imbalanced data distributions, or imbalanced data sets. In other words, in these problem domains the cases that are more important for the user are rare and poorly represented in the population and the available training set.

Let the user preference bias be expressed by an importance or relevance function $\phi()$ that maps the values of the target variable into a range of importance, where 1 is maximal importance and 0 minimum relevance,

$$\phi(Y) : \mathcal{Y} \rightarrow [0, 1] \tag{2.1}$$

where $\mathcal{Y}$ is the domain of the target variable Y.

Suppose the user defines a relevance threshold t_R which sets the boundary above which the target variable values are relevant for the user. Let $D_R \in D$ be the subset of the training samples for which the relevance of the target variable values is above the defined threshold t_R, and $D_N \in D$ be the subset of the training sample with the normal, i.e. less important, cases for the user,

$$D_R = \{\langle \mathbf{x}_i, y_i \rangle \ \in D : \phi(y_i) > t_R\} \tag{2.2}$$

$$D_N = \{\langle \mathbf{x}_i, y_i \rangle \ \in D : \phi(y_i) \leq t_R\} = D \setminus D_R \tag{2.3}$$

The problem of imbalanced data sets can be described by the following assertions:

- $\phi(Y)$ is not uniform across the domain of Y
- The cardinality of the set of examples D_R is much smaller than the cardinality of D_N
- The used evaluation criteria for both learning the models and evaluating their performance assumes an uniform $\phi(Y)$, i.e. it is insensitive to $\phi(Y)$.

In this type of tasks we are facing a situation where the obtained models are sub-optimal with respect to the user-preference biases and, moreover, the metrics used to evaluate them are not in accordance with these biases and thus may be misleading. Given the above-described properties of these predictive tasks we face two main challenges:

- the definition of special purpose evaluation metrics that are biased towards the performance of the models on the cases in D_R, and

- the development of strategies for getting the learning algorithms to focus on the cases in D_R.

These two challenges must be addressed, otherwise, the built models will tend to be biased on the most frequent (and less interesting for the user) cases, and the evaluation results will not capture the competence of the models on the relevant cases. Regarding proper evaluation, several metrics have been proposed, mainly for classification tasks, to overcome the difficulties of traditional metrics that are no longer adequate as they do not take into account the user preferences. At the modelling level, a large number of solutions that try to make the models focus on the less frequent and more important cases for the user were proposed for classification tasks.

The large number of contributions made within the classification setting led to the emergence of a specific vocabulary. These terms, although not suitable for regression tasks, will be used whenever classification tasks are mentioned. For instance, when the target variable Y is nominal, the imbalanced domain problem is usually referred to as the class imbalance problem or the *between-class* imbalance problem. The last expression highlights the existing unbalance among the different classes of the domain. Also, the previously defined D_R set containing the rare and more relevant cases for the user is traditionally called the minority or positive class. The set D_N with the less important cases for the user and the more frequent ones is named the majority or negative class. The concept of imbalance ratio (or class-imbalance ratio for nominal Y) is used to refer to the ratio of D_N to D_R.

2.2 Related Problems

The imbalanced data distribution is regarded as a major obstacle for predictive modelling in the presence of a user preference bias towards the least represented examples. Nevertheless, other problems exist that may also degrade the models performance and frequently coexist with the imbalanced domain problem.

These related problems have been addressed mainly within a classification setting. Problems such as small disjuncts, class overlap and small sample size, usually coexist with imbalanced classification domains and are also identified as possible causes of classifiers performance degradation (Weiss, 2004; He and Garcia, 2009; Sun et al., 2009). We will briefly describe the major developments in the context of the following related problems:

1. class overlapping or class separability,
2. small sample size and lack of density in the training set,
3. high dimensionality of the data set,
4. noisy data,
5. small disjuncts or data fragmentation.

The overlap problem occurs when a given region of the data space contains an identical number of training cases for each class. In this situation, a learner will have an increased difficulty in distinguishing between the classes present on the overlapping region. The problems of imbalanced data sets and overlapping regions were mostly treated separately. However, in the last decade some attention was given to the relationship between these two problems in the performance degradation of classifiers (Prati et al., 2004a; García et al., 2006b). The combination of imbalanced domains with overlapping regions causes an important deterioration of the learner performance and both problems acting together produce much more difficulties than expected when considering their effects in isolation (Denil and Trappenberg, 2010). Several strategies for addressing both problems simultaneously have been developed. Recent works (Alejo Eleuterio et al., 2011; Alejo et al., 2013) present combinations of solutions for handling, simultaneously, both the class imbalance and the class overlap problem and apply a blend of techniques for addressing these issues. For instance, the proposal of Alejo Eleuterio et al. (2011) uses editing techniques and a modification in the mean square error cost function for a multilayer Perceptron and the approach of Alejo et al. (2013) applies a Gabriel graphs editing technique to address the overlapping classes by removing noisy and borderline negative samples, and a modified back-propagation algorithm to deal with imbalanced classes.

The imbalance problem and the overlapping of regions are considered in García et al. (2007) with an additional difficulty: the overall imbalance ratio is different from local imbalance ratios in overlaping regions. In this case the task of learning becomes a major challenge. A similar setting is studied in García et al. (2008c) where artificial data sets are used to generate overlapping regions with an imbalance ratio inverse to the overall imbalance ratio of the data set. In these particular conditions, the more represented class in overlap regions tends to be better classified by methods based on global learning, while the class less represented in such regions tends to be better classified by local methods.

The small training set, or small sample problem, is also naturally related to the imbalanced domain problem. In an imbalanced context, having too few examples in the set D_R (the relevant and rare examples, or the minority class) will prevent the learner of satisfyingly capture their characteristics and will hinder the generalization capability of the algorithm. The relation between imbalanced domains and small sample problems was addressed in Japkowicz and Stephen (2002) and Jo and Japkowicz (2004) where it is highlighted that class imbalance degrades classification performance in small data sets although this loss of performance tends to gradually reduce as the training set size increases. As expected, the subconcepts defined by the minority class examples can be better learned if their number can be increased.

The small sample problem may trigger problems such as rare cases (Weiss, 2005), which cover only a few training examples, and so bring an additional difficulty to the learning system. Rare examples are extremely scarce cases which presence is associated with the problem of lack of data. These examples are difficult to detect and, when they are detected, it is extremely difficult to make a generalisation from only a few data samples. The small training set problem may also be accompanied with other problems as variable training class distribution, i.e., a variable class distribution which may not match the target distribution. In many real world problems the class distribution of the training set is often diverse, unknown in advance, and does not match the testing or target distributions, which may also vary over time. In Forman and Cohen (2004) it is shown that, for imbalanced domains, obtaining a balanced training set is not the most favourable setting and classifiers performance can be greatly improved by non-random sampling that favours the minority class.

For domains as text classification, web categorization and biological/medical data, the imbalance problem is usually accompanied with high dimensional data sets. In such setting, the user is interested in a rare and more important class which is present in a data set with a high number of predictors (Chawla et al., 2004). The main challenge here is to adequately select features that contain the key information of the problem. Feature selection is recommended (Wasikowski and Chen, 2010) and is also pointed as the solution for addressing the class imbalance problem (Mladenic and Grobelnik, 1999; Zheng et al., 2004; Chen and Wasikowski, 2008; Van Der Putten and Van Someren, 2004; Forman, 2003). Several proposals exist for handling the imbalance problem in conjunction with the high dimensionality problem, all using a feature selection strategy (Zheng et al., 2004; Del Castillo and Serrano, 2004; Forman and Cohen, 2004; Chu et al., 2010). For instance, in Zheng et al. (2004) it is suggested that the existing measures used for feature selection are not very appropriate for imbalanced domains. Thus, a new feature selection framework is proposed, which selects features for positive and negative classes separately and then explicitly combines them.

Noise is a known factor that usually affects models performance. In imbalanced domains, noisy data has a greater impact on the least represented examples (Weiss, 2004). A recent study (Seiffert et al., 2011) on the effect of noise used the software quality data domain which is intrinsically characterised by the presence of class imbalance and class noise. It was concluded that, generally, class noise has a more significant impact on learners than imbalance. The used data sets had the characteristic of as the level of noise decreased, the imbalanced was increased, and so, it was observed that the reduction of noise improved the sampling techniques performance although the imbalanced increased simultaneously. However, at the highest level of imbalance the performance dropped. It is also noticed that the interaction between the level of imbalance and the level of noise within a data set is a significant factor, and that studying these two main effects in isolation may not be sufficient.

Although the *between-class* imbalance is more widely known, another type of imbalance exists: the *within-class* imbalance which is the imbalance occurring between the subclusters of each class in the data set (Japkowicz, 2001a; Jo and Japkowicz, 2004). This second type of imbalance is not quite as well known or extensively studied as the *between-class* imbalance is.

The *within-class* imbalance problem along with the *between-class* imbalance problem are instances of the general problem known as the problem of small disjuncts (Japkowicz, 2001a). Systems learning from examples do not usually create a purely conjunctive definition of each concept. They generate a definition made up of several disjuncts, where each disjunct is a conjunctive definition of a subconcept of the original concept. The *coverage* of a disjunct is defined as the number of training examples it correctly classifies. A disjunct is called small if it has a low *coverage* (Holte et al., 1989), i.e., it classifies few training examples.

Small disjuncts are a problem due to the tendency of classification methods to overfit and misclassify these examples since the learners are typically biased towards classifying large disjuncts. The following reasons are pointed for considering small disjuncts a problem:

- many concepts include rare or exceptional cases and it is important for induced definitions to learn from these cases;
- small disjuncts are a significant portion of an induced definition, i.e., they

collectively match a significant percentage of the examples in a definition;

- small disjuncts have a much higher error rate than large disjuncts, collectively contributing to a significant portion of the total errors (Weiss, 2010).

Regarding all the previously mentioned related problems, the relationship between the problem of class imbalance and the problem of small disjuncts is the most studied and much attention has been given to the small disjuncts problem. This problem is often present along with the problem of class imbalance in real world data sets and the connection existing between the two problems is not yet well understood (Jo and Japkowicz, 2004). In fact, several works exist which address the problem of small disjuncts and the class imbalance problem. Works as Japkowicz (2003); Weiss and Provost (2003); and Jo and Japkowicz (2004) refer to small disjuncts as the main responsible for performance loss, although recognising that they can be a consequence of the presence of rare cases, domains with a small training set size and high complexity settings. On the other hand, in some domains the class imbalance problem is apparently more relevant than the problem of small disjuncts. This is suggested, for instance, in Pearson et al. (2003). Even in the experiences conducted in Jo and Japkowicz (2004), although the majority of the experiences in artificial domains point to the small disjuncts as the cause of degradation of classifiers performance, a specific domain exists that points in the opposite direction. So, further research is necessary to evaluate which conditions make a domain more or less sensitive to class imbalances than to small disjuncts (Jo and Japkowicz, 2004).

For studying the impact of small disjuncts a new metric called *error concentration* was defined in Weiss and Hirsh (2000) for expressing the error concentration towards the smaller disjuncts. The work in Weiss (2010) analyses the impact of several factors on small disjuncts and in the error distribution across disjuncts. Among the studied factors are pruning, training-set size, noise and class imbalance. In this work, pruning is analysed as a strategy for addressing the problem of small disjuncts, and it is concluded that it redistributes the errors more uniformly. However, in the context of imbalanced domains, this is exactly the opposite of the intended behaviour, as it is more important to classify a reduced set of examples with high precision than finding the classifier with the best overall accuracy. Thus, pruning is not considered effective for dealing with small disjuncts in the presence of class imbalance (Prati et al., 2004b; Weiss, 2010).

Given that previous studies concluded that the disjunct size was part of the reason for minority class predictions to be more error prone, in Weiss (2010) the existence of a link in the opposite direction is studied. One of the conclusions is that, even with a balanced data set, errors tend to be concentrated towards the smaller disjuncts. However, when there is class imbalance, the error concentration increases. Those differences tend to be larger when the data set has greater class imbalance. Thus, class imbalance is partly responsible for the problem with small disjuncts, and artificially modifying the class distribution of the training data to be more balanced, causes a decrease in the error concentration.

With this notion of a possible connection between *within-class* and *between-class* imbalance problems several proposals were made which address simultaneously both problems (Japkowicz, 2001a; Jo and Japkowicz, 2004; Prati et al., 2004b).

Some recent works as Napierała et al. (2010) study the impact of borderline and noisy examples on the classifier performance. The number of minority class borderline examples is found to strongly affect the classifier performance. Moreover, the authors relate the performance of existing strategies for addressing the class imbalance problem with the amount of overlapping area in the data set and the existence of noisy majority class examples.

Chapter 3

Performance Assessment Metrics for Imbalanced Domains

3.1 Introduction

Obtaining a model from data can be seen as a search problem guided by an evaluation criterion that establishes a preference ordering among different alternatives. The main problem of imbalanced data sets lies on the fact that this is often associated with a user preference bias towards cases that are poorly represented in the available data sample. Standard evaluation criteria tend to focus the evaluation of the models on the most frequent cases, which may be against the user preferences. In fact, the use of common metrics in imbalanced domains might produce misleading conclusions since they are insensitiveto skew domains (Ranawana and Palade, 2006; Daskalaki et al., 2006). As such, selecting proper evaluation metrics plays a key role in the task of correctly handling data imbalance. Adequate metrics should not only provide means to compare the models according to the user preferences, but can also be used to drive the learning of these models by biasing the algorithms for the models that the user prefers.

As the problem of imbalanced domains has been addressed mainly for classification tasks, there are far more solutions regarding performance metrics for these tasks than for regression tasks . We start by addressing the problem of evaluation metrics in classification problems (Section 3.2) and then move to regression tasks (Section 3.3).

3.2 Metrics for Classification Tasks

Typically, *accuracy* (cf. Equation 3.1) and its complement *error rate* (cf. Equation 3.2) are the most frequently used metrics for estimating the performance of learning systems in classification problems. For two classes problems, these metrics can be defined as follows,

$$accuracy = \frac{TP+TN}{TP+FN+TN+FP} \tag{3.1}$$

$$error \quad rate = 1 - accuracy \tag{3.2}$$

Considering a two-class problem, the confusion matrix (or contingency table) presents the results of correctly and incorrectly recognised examples of each class (cf. Table 3.1). This table provides the number of True Positive (TP) and True Negative (TN), i.e. the instances that were correctly classified for each class, and the number of False Positive (FP) and False Negative (FN), i.e. the type I and type II errors.

		Predicted	
		Positive	Negative
True	Positive	TP	FN
	Negative	FP	TN

Table 3.1: Confusion matrix for a two-class problem.

Considering a user preference bias towards the minority class examples, *accuracy* is not suitable because the impact of the least represented examples is reduced when compared to that of the majority class. As an example, consider a problem where the minority class, is represented only by 1% of the training examples. To achieve an *accuracy* of 99% it is enough to predict, for every example, the majority class label. Yet, all the minority examples, the more interesting and relevant for the user, are misclassified. When the concern is the identification of the rare cases these metrics are clearly inappropriate.

As mentioned before, in the context of imbalanced domains, the use of common metrics as *accuracy* can lead to sub-optimal classification models (He and Garcia, 2009; Weiss, 2004; Kubat and Matwin, 1997). The used metrics must consider the user preferences and,thus, should take into account the data distribution. To fulfill this goal it became necessary to develop and use alternative performance measures. From Table 3.1 the following measures (cf. Equations 3.3-3.8) can be obtained,

$$true\ positive\ rate\ (recall\ or\ sensitivity):\ TP_{rate} = \frac{TP}{TP+FN} \tag{3.3}$$

$$true\ negative\ rate\ (specificity\ ):\ TN_{rate} = \frac{TN}{TN+FP} \tag{3.4}$$

$$false\ positive\ rate:\ FP_{rate} = \frac{FP}{TN+FP} \tag{3.5}$$

$$false\ negative\ rate:\ FN_{rate} = \frac{FN}{TP+FN} \tag{3.6}$$

$$positive\ predictive\ value\ (precision\ ):\ PP_{value} = \frac{TP}{TP+FP} \tag{3.7}$$

$$negative\ predictive\ value:\ NP_{value} = \frac{TN}{TN+FN} \tag{3.8}$$

Using one of these measures (Equation 3.3 to Equation 3.8) alone is still not adequate. The user would have to monitor the results of multiple metrics separately. Given that simultaneously monitoring two metrics is impractical, different proposals arose for combining individual measures as the *F-measure* (Estabrooks and Japkowicz, 2001), the *geometric mean* (Kubat et al., 1998) or the *receiver operating characteristic* (*ROC*) curve (Bradley, 1997).

The *F-Measure* (F_β), a combination of both *precision* and *recall*, is defined as follows:

$$F_\beta = \frac{(1+\beta)^2 \cdot recall \cdot precision}{\beta^2 \cdot recall + precision} \tag{3.9}$$

where β is a coefficient to adjust the relative importance of *recall* with respect to *precision* (if $\beta = 1$ *precision* and *recall* have the same weight).

Precision is a measure of exactness, assessing how many of the examples labelled as positive are actually correctly labelled. On the other hand, *recall* is a measure of completeness, expressing how many examples of the positive class are correctly labelled. F_β is commonly used and is more informative about the effectiveness of a classifier on predicting correctly the cases that matter to the user. This metric value is high when both *recall* and *precision* are high .

An equally important metric is the *geometric mean* (*G-Mean*) which is defined as:

$$G - Mean = \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{TN + FP}} = \sqrt{sensitivity \times specificity} \tag{3.10}$$

G-Mean computes the *geometric mean* of the accuracies of the two classes, attempting to maximize them while obtaining good balance.

When dealing with imbalanced data sets, one of the most popular tools is the *receiver operating characteristics* (*ROC*) curve and the associated use of the area under the *ROC* curve (*AUC*). This approach plots the *true positive rate* (cf. Equation 3.3) on the X axis over the *false positive rate* (cf. Equation 3.5) on the Y axis. A point in *ROC* space corresponds to the performance of a given classifier on a certain distribution. A ROC curve provides information for all the values of a decision/threshold parameter for classifying an example as belonging to a given class.

The usefulness of the *ROC* curve is the visualization of the relative trade-off between the benefits (TP_{rate}) and costs (FP_{rate}) of classification regarding data distributions. The ideal model would obtain $TP_{rate} = 1$ and $FP_{rate} = 0$, thus a good model should be as closer as possible to $(1, 0)$ point. On the other hand, a random model should remain along the main diagonal, connecting the points $(0, 0)$ and $(1, 1)$, which represent that all predictions are from the negative class and from the positive class respectively (cf. Figure 3.1). Thus, any classifier that lies on the lower right triangle performs worse than random guessing.

Comparing several models through *ROC* curves is not an easy task unless one of the curves clearly dominates all the others over the entire space (Provost and Fawcett, 1997). Not delivering a single performance measure is a clear disadvantage of *ROC* curves. The *AUC* measure, which is determined by calculating the area under the *ROC* graphic (Equation 3.11), provides one single measure allowing the evaluation of the best model on average. Still, it is not biased towards the minority class.

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} = \frac{TP_{rate} + TN_{rate}}{2} \tag{3.11}$$

When data sets are highly skewed, *precision-recall curves* (*PR curves*) may be preferred over *ROC* curves as the later may lead to an excessively optimistic view of

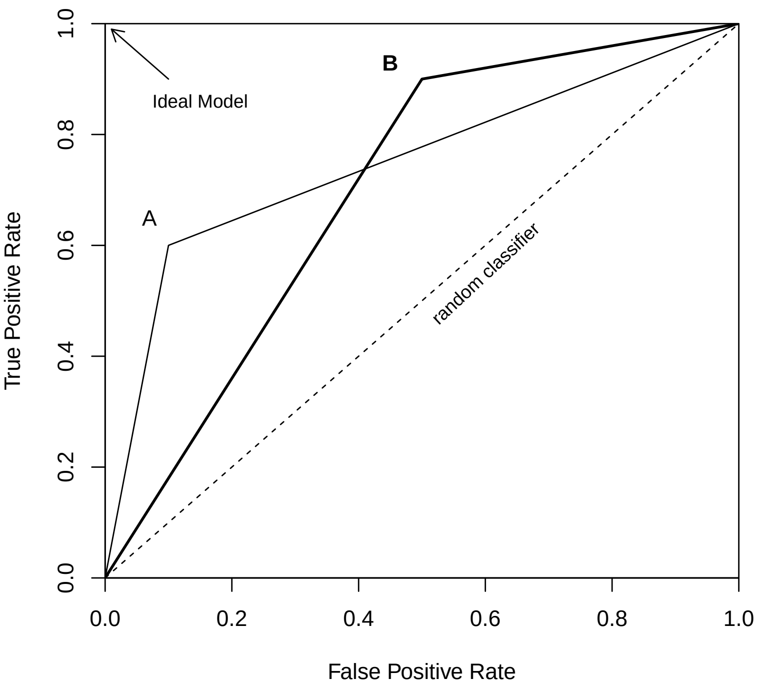


Figure 3.1: *ROC* curve of three classifiers: A, B and random.

the algorithm performance. On this case, *PR curves* are recommended for providing a more informative representation of performance assessment (Davis and Goadrich, 2006). A *PR curve* plots the *recall* rate on the X axis over the *precision* rate on the Y axis. There is a strong relation between these two curves: a curve dominates in *ROC* space if and only if it dominates in *PR* space (Davis and Goadrich, 2006). However, it is also shown in Davis and Goadrich (2006) that an algorithm that optimizes the area under the *ROC* curve is not guaranteed to also optimize the area under the *PR* curve.

AUC and *G-Mean* have a known drawback: they provide exactly the same result for many different combinations of True Positive Rate and True Negative Rate. Also, they are unable to reflect each class contribution to the overall performance and do not identify which is the prevalent class. To deal with the *AUC* and *G-Mean* inability to explain the contribution of each class to the overall performance, a new metric called *dominance* is proposed in García et al. (2008b) which is defined as:

$$dominance = TP_{rate} - TN_{rate} \tag{3.12}$$

This measure ranges from -1 to $+1$, where a value of $+1$ represents a situation of perfect *accuracy* on the positive class, but failing on all negative cases, while a value of -1 corresponds to the opposite situation. Individual rates are perfectly balanced if $dominance = 0$.

The *index of balanced accuracy* (*IBA*) (García et al., 2009; Garcia et al., 2010) quantifies a trade-off between an index of how balanced both class accuracies are and a chosen unbiased measure of overall *accuracy*. This metric aims to favour classifiers with better results on the positive class thus being more sensitive to imbalanced data sets. *IBA* measure is defined as:

$$IBA_{\alpha}(M) = (1 + \alpha \cdot dominance)M \tag{3.13}$$

where $(1 + \alpha \cdot dominance)$ is the weighting factor and M represents any performance metric. *IBA* is strongly correlated with AUC and *G-Mean*. However, unlike these metrics, *IBA* is positively correlated with TP_{rate} and negatively correlated with *accuracy*.

Other measures have been proposed, as the *optimized precision* (Ranawana and Palade, 2006) which is defined as:

$$optimized\ precision = accuracy - \frac{|TN_{rate} - TP_{rate}|}{TN_{rate} + TP_{rate}} \tag{3.14}$$

High values of *optimized precision* are obtained with high global *accuracy* and well balanced class accuracies. Nevertheless, this measure can be strongly affected by the bias of the global *accuracy*.

The *adjusted geometric mean* (*AG-Mean*) (Batuwita and Palade, 2009, 2012) was proposed to overcome some problems identified in F_{β}, AUC and *G-Mean* which are related with the changes in *sensitivity* (cf. Equation 3.3) and *specificity* (cf. Equation 3.4). Under imbalanced domains it is usual to apply a method which produces an increase in *sensitivity* by sacrificing some amount of *specificity*. However, in some domains, it is important to improve the *sensitivity* as much as possible while keeping the reduction in *specificity* to the minimum. Therefore, *AG-Mean* was built with the aim of being more sensitive to changes in *specificity* than to changes in *sensitivity* and also to incorporate a dependence on the proportion of the majority class examples in the data set. Thus, the higher the imbalance the higher the sensitiveness of the measure to the changes in *specificity*. The *AG-Mean* is defined as:

$$AG - Mean = \begin{cases} \frac{G-Mean+specificity \cdot N_n}{1+N_n} & if\ sensitivity > 0 \\ 0 & if\ sensitivity = 0 \end{cases} \tag{3.15}$$

where N_n is the proportion of majority class examples in the dataset.

To tackle the problem detected in *AUC* measure of implicitly using different misclassification cost distributions for different classifiers, the *H-measure* was developed (Hand, 2009). This measure uses a symmetric Beta distribution to replace the implicit cost weight distribution in the *AUC*.

However, in the context of imbalanced domains *H-measure* is still not adequate as it equally penalises errors made on the positive and negative class. Thus, this metric is more suitable for balanced domains. The need for considering different weights for mistakes made on different classes, lead to the development of a new metric called *B42* which was proposed by Thai-Nghe et al. (2011). In fact, under an imbalanced domain, misclassifying a minority class example is much more serious than misclassifying a majority class example. The *B42* metric also replaces the cost weight distribution assumed in *AUC* metric, but adopts an asymmetric Beta distribution for penalising more the errors made on the minority class. The Beta distribution chosen was $Beta(x, 4, 2)$ although the authors refer that other asymmetric Beta distributions could have also been selected for this purpose.

3.3 Metrics for Regression Tasks

Unlike classification problems, very few efforts have been made regarding evaluation metrics for regression tasks in imbalanced domains. Performance measures commonly used in regression, such as *Mean Squared Error* (MSE) and *Mean Absolute Deviation* (MAD), Equations 3.16 and 3.17, are not adequate to regression problems in imbalanced domains. In fact, they presume an uniform user preference bias over the domain and take all the prediction errors equally across the domain of the target variable, assuming that the magnitude of the committed error is the decisive factor for the cost assigned to a prediction.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.16}$$

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.17}$$

However, although the magnitude of the numeric error is important, it is also important where the error has occurred, i.e. the error metric must also be sensitive to the location of the errors within the target variable range.

Supposing a user preference bias towards the rare extreme values, one possible way to overcome this problem would be to consider a weighted error measure, such that higher weights are given to the rare extreme values cases (cf. Equation 3.18).

$$Err_w = \frac{\sum_{i=1}^{n} w_i \cdot L(y_i, \hat{y}_i)}{\sum_{i=1}^{n} w_i} \tag{3.18}$$

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

However, this solution would only take into account one part of the problem. In fact, the metric Err_w considers the errors of bad predictions for relevant values, but fails to consider the reverse, neglecting the errors of predicting a rare value when it is a normal one (Ribeiro, 2011).

In the context of financial applications (Christoffersen and Diebold, 1996; Crone et al., 2005), the issue of differentiated prediction costs was addressed and asymmetric linear loss functions were proposed. The *LIN-LIN* error metric (cf. Equation 3.19) aims at distinguishing two type of errors: under-predictions ($\hat{y} < y$) and over-predictions ($\hat{y} > y$).

$$LIN - LIN = \begin{cases} c_o|y - \hat{y}| & if\ \hat{y} > y; \\ 0 & if\ \hat{y} = y; \\ c_u|y - \hat{y}| & if\ \hat{y} < y. \end{cases} \tag{3.19}$$

This metric allows to differentiate the errors depending on where they occur: if c_o and c_u are different, two errors with the same amplitude, occurring in different "sides", will have different penalisation.

Nevertheless, *LIN-LIN* metric only distinguishes between these two situations (under- and over-predictions). Moreover, this measure considers all under- (over-) predictions as equally serious, taking only into account the error magnitude as in standard error metrics. Thus, this approach is still not adequate for imbalanced data sets having a non uniform user preference bias across the target variable domain (Ribeiro, 2011).

Besides the *LIN-LIN* metric, which is asymmetric linear, many different kinds of asymmetric loss functions have been explored: $QUAD\text{-}QUAD$ (asymmetric quadratic), *LINEX* (approximately linear on one side and exponential on the other side), *double LINEX* (aims at making *LINEX* more flexible) and *QUAD-EXP* (approximately quadratic on one side and exponential on the other side) (Zellner, 1986; Cain and Janssen, 1995; Christoffersen and Diebold, 1996, 1997; Crone et al., 2005; Granger, 1999; Lee, 2008). However, they all suffer from the same problem as *LIN-LIN* metric: they only distinguish over-predictions from under-predictions. Thus, they are still not adequate for the problem of imbalanced domains with a user preference bias towards some specific values.

Following the efforts made within classification, some attempts were made to adapt the existing notion of ROC curves to regression tasks. One of these attempts is the ROC space for regression (RROC space) (Hernndez-Orallo, 2013) which is motivated by the asymmetric loss often present on regression applications where over-estimations are not equally costly as under-estimations (or vice versa). RROC space is defined by plotting the total over-estimation on the X axis and the total under-estimation on the Y axis (cf. Figure 3.2). RROC curves are obtained when the notion of shift is used, which is a constant that can be added (or subtracted) to example predictions in order to adjust the model to an asymmetric operating condition. The notion of dominance can also be assessed by plotting different regression models, similarly to ROC curves in classification problems.

Other evaluation metrics were explored, such as the area over the RROC curve (AOC) which was shown to be equivalent to the error variance.

In spite of the importance of this approach, it still only distinguishes over predictions from under predictions and, as we have mentioned before, this is not enough in the context of imbalanced domains with a non uniform user preference bias over the target variable. So, it is also important to consider where the errors occurred over the target variable range.

Another relevant effort towards the adaptation of the concept of ROC curves to regression tasks was made by Bi and Bennett (2003) with the proposal of Regression Error

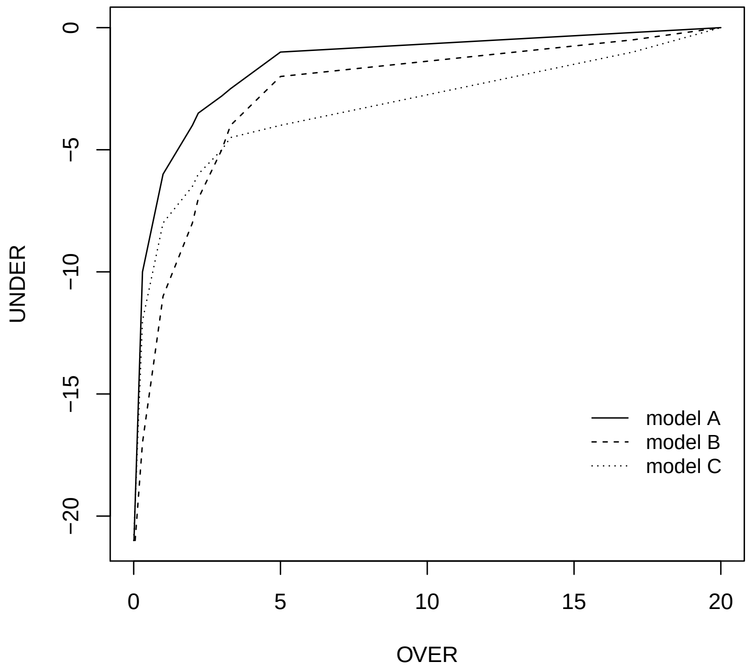


Figure 3.2: *RROC* curve of three models: A, B and C.

Characteristic (REC) curves which provide a graphical description of the cumulative distribution function (cdf) of the error of a model. On these curves, the error tolerance is plotted on the X axis and on the Y axis is plotted the accuracy of a regression function which is defined as the percentage of points predicted within a given tolerance ϵ:

$$cdf(\epsilon) = \frac{|(\mathbf{x}_i, y_i) : L(\hat{y}_i, y_i) \leq \epsilon,\ i = 1, \ldots, m|}{m} \tag{3.20}$$

where m is the total number of data points. REC curves illustrate the predictive performance of a model across the range of possible errors.

As with ROC curves, it is possible to represent several models in the same space being possible to determine dominance regions (cf. Figure 3.3). A model dominates another if its REC curve is always above the other models curve. It can also be calculated the Area Over the Curve (AOC) which is a biased estimate of the expected error of a model (Bi and Bennett, 2003).

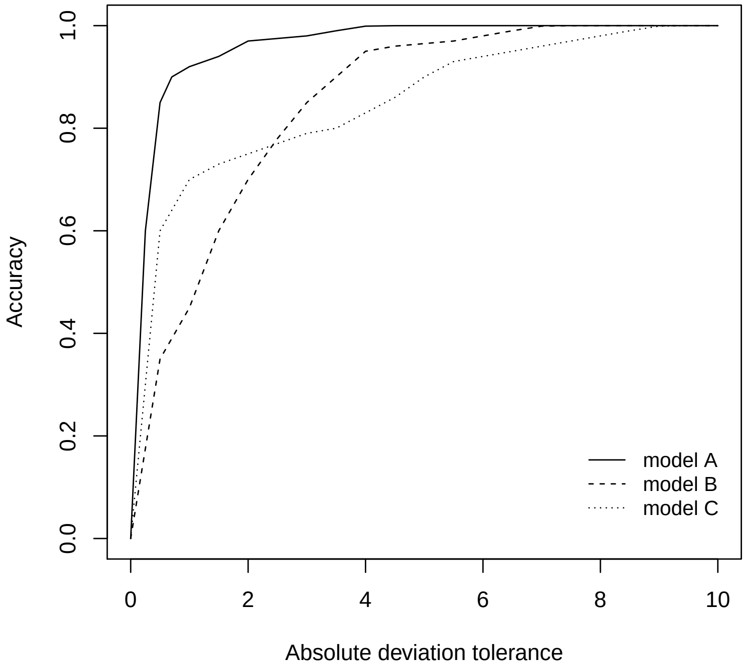


Figure 3.3: *REC* curve of three models: A, B and C.

Although having several advantages, REC curves are still not adequate to imbalanced domains in the presence of a user preference bias towards some specific target values. In this case, the same error amplitude can have a different importance to the user depending on the true target variable value. In fact, we could have two different models with the same REC curves but one being preferred over the other based on the errors made on target values that are more relevant to the user. So, it is also essential to inspect the errors over the target variable domain.

To address this problem Regression Error Characteristic Surfaces (RECS) were proposed by Torgo (2005). REC surfaces are an extension of REC curves where the cumulative distribution of the dependent variable is set as an additional dimension. RECS show how the errors corresponding to a certain point of the REC curve are distributed across the range of the target variable. Figures 3.3 and 3.4 show an example of REC curves and a REC surface. REC surfaces are quite relevant and useful in the context of imbalanced domains combined with a user preference bias towards some specific target values. In fact, it is important to study the performance of the models as a function of the target variable range. This tool allows the study of

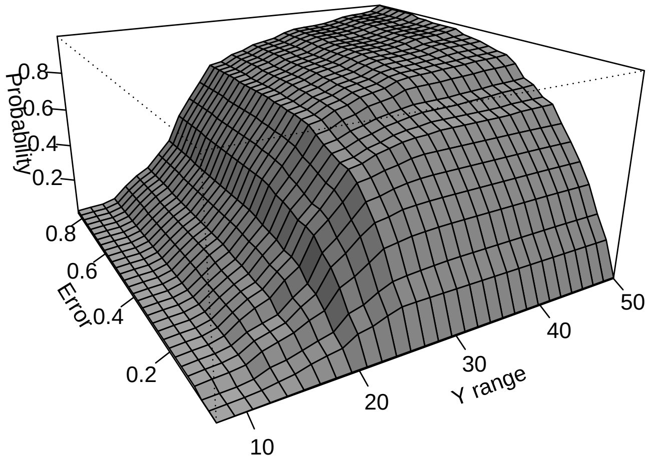


Figure 3.4: An example of the *REC* surface.

the behaviour of alternative models for certain specific values of the target variable. For instance, the performance over the values that are more relevant for the user can be inspected. The user can also establish a certain range of errors, assessing afterwards in which parts of the target variable range they are more frequent.

Another existing approach is the precision/recall evaluation framework, based on the concept of utility-based regression (Ribeiro, 2011; Torgo and Ribeiro, 2007). At the core of utility-based regression is the notion of relevance of the target variable values and the assumption that this relevance is not uniform across the domain of this variable. This notion is motivated by the fact that, contrary to standard regression, in some domains, as imbalanced domains, not all the values are equally important/relevant. In utility-based regression the usefulness of a prediction is a function of both the numeric error of the prediction (given by some loss function $L(\hat{y}, y)$) and the relevance (importance) of both the predicted $\hat{y}$ and true y values. Relevance is the crucial property that expresses the domain-specific biases concerning the different importance of the values. As we have mentioned it is defined as a continuous function $\phi(Y) : \mathcal{Y} \rightarrow [0, 1]$ that maps the target variable domain $\mathcal{Y}$ into a

$[0,1]$ scale of relevance, where 0 represents the minimum and 1 represents the maximum relevance.

Being a domain-specific function, it is the user responsibility to specify the relevance function. However, Ribeiro (2011) describes some specific methods to automatically obtain these functions when the goal is to be accurate at rare extreme values. The methods are based on the simple observation that, for these applications, the notion of relevance is inversely proportional to the target variable probability density function. Figure 3.5 shows the relevance function $\phi()$ automatically generated for the $a1$ data set using the mentioned methods.

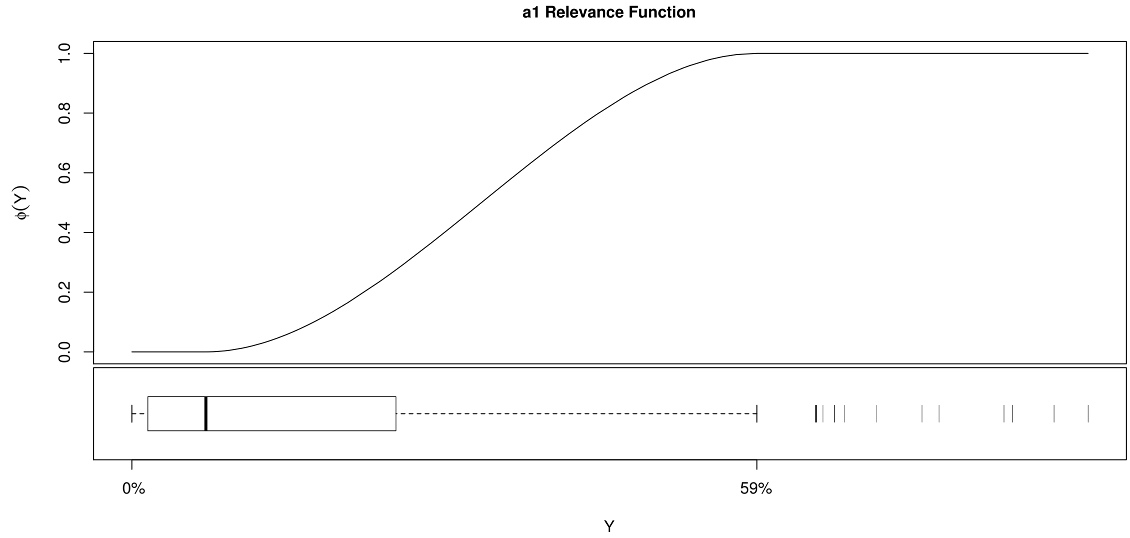


Figure 3.5: Relevance function $\phi()$ automatically generated for the *a1* data set.

For the particular subset of applications associated with rare extreme values, the utility of a model prediction is related to the question on whether it has led to the identification of the correct type of extreme (high or low) and if the prediction was precise enough in numeric terms. Thus, to calculate the utility of a prediction it is necessary consider two aspects: (i) does it identify the correct type of extreme? (ii) what is the numeric accuracy of the prediction (i.e. $L(\hat{y}, y)$)? This latter issue is important because it allows for coping with different "degrees" of actions as a result of the model predictions. For instance, in the context of financial trading, an agent may use a decision rule that implies buying an asset if the predicted return is above a certain threshold. However, this same agent may invest different amounts depending on the predicted return, and thus the need for precise numeric forecasts of the returns on top of the correct identification of the type of extreme. This numeric precision, together with the fact that we may have more than one type of extreme (i.e. more than one "positive" class) are the key distinguishing features of this framework when compared to pure classification approaches.

The concrete utility score of a prediction, in accordance with the original framework of utility-based learning (e.g. Elkan (2001); Zadrozny (2005)), results from the net balance between its benefits and costs (i.e. negative benefits). A prediction should be considered beneficial only if it leads to the identification of the correct type of extreme. However, the reward should also increase with the numeric accuracy of the prediction and should be dependent on the relevance of the true value. In this context, Ribeiro (2011) has defined the notions of benefits and costs of numeric predictions, and proposed the following definition of the utility of the predictions of a regression model,

$$\begin{aligned} U_\phi^p(\hat{y}, y) &= B_\phi(\hat{y}, y) \; - \; C_\phi^p(\hat{y}, y) \\ &= \phi(y) \cdot (1 - \Gamma_B(\hat{y}, y)) \; - \; \phi^p(\hat{y}, y) \cdot \Gamma_C(\hat{y}, y) \end{aligned} \tag{3.21}$$

where $B_\phi(\hat{y}, y)$, $C_\phi^p(\hat{y}, y)$, $\Gamma_B(\hat{y}, y)$ and $\Gamma_C(\hat{y}, y)$ are functions related to the notions of costs and benefits of predictions that are defined in Ribeiro (2011). Figure 3.6 shows the utility isometrics and the utility surface for the *a1* data set considering that the false alarms are not relevant.

Precision and recall are two of the most commonly used metrics to estimate the performance of models in highly skewed domains (Davis and Goadrich, 2006). The notions of precision and recall were adapted to regression problems with non-uniform relevance of the target values by Torgo and Ribeiro (2009) and Ribeiro (2011). These metrics are usually defined as ratios between the correctly identified events (usually known as true positives within classification), and either the signalled events (for precision), or the true events (for recall). Ribeiro (2011) defines the notion of event using the concept of utility. In this context, the ratios of the two metrics are also defined as functions of utility, finally leading to the following definitions of precision and recall for regression,

$$recall = \frac{\sum\limits_{i:\hat{z}_i=1, z_i=1} (1 \, + \, u_i)}{\sum\limits_{i:z_i=1} (1 \, + \, \phi(y_i))} \tag{3.22}$$

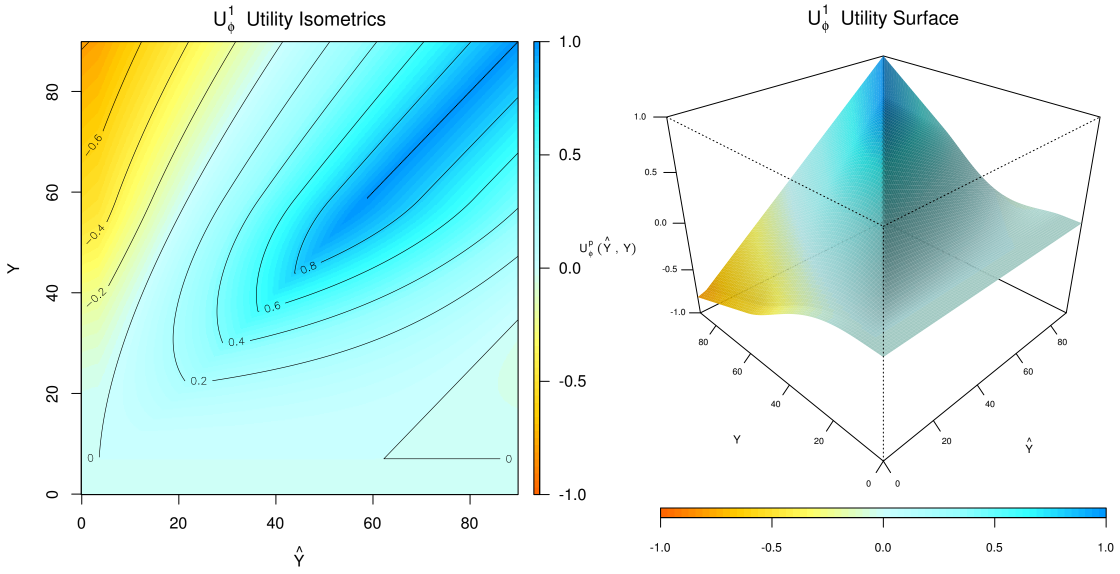


Figure 3.6: Utility surface for the *a1* data set obtained with relevance function $\phi()$ shown in Figure 3.5

and

$$precision = \frac{\sum\limits_{i:\hat{z}_i=1,z_i=1} (1 + u_i)}{\sum\limits_{i:\hat{z}_i=1,z_i=1} (1 + \phi(y_i)) + \sum\limits_{i:\hat{z}_i=1,z_i=0} (2 - p\,(1 - \phi(y_i)))} \tag{3.23}$$

where p is a weight differentiating the types of errors, while $\hat{z}$ and z are binary properties associated with being in the presence of a rare extreme case[1].

We propose an alternative definition for precision and recall, which is also based on the utility-based framework defined by Ribeiro (2011). This formulation also assumes an user-defined threshold of relevance, t_R, which is used for distinguishing cases which are signalled events ($\phi(\hat{y}_i) > t_R$) or true events ($\phi(y_i) > t_R$) from the normal and irrelevant cases for the user. The key difference of this proposal is the identification of signalled/true events which is solely dependent on the relevance function (domain knowledge) and not on the utility of the predictions made by the model.

Therefore, we propose the following alternative definitions of precision and recall for regression,

[1] Full details can be obtained in Chapter 4 of Ribeiro (2011).

$$recall = \frac{\sum\limits_{\phi(y_i)\, >\, t_R} (1\, +\, u_i)}{\sum\limits_{\phi(y_i)\, >\, t_R} (1\, +\, \phi(y_i))} \tag{3.24}$$

and

$$precision = \frac{\sum\limits_{\phi(\hat{y}_i)\, >\, t_R} (1\, +\, u_i)}{\sum\limits_{\phi(\hat{y}_i)\, >\, t_R} (1\, +\, \phi(\hat{y}_i))} \tag{3.25}$$

Having this formulation, and similarly to what is done in classification, the *F-measure* (cf. Equation 3.9) can be obtained based on the previous definitions of precision and recall.

In this thesis, we will evaluate the proposed models using these definitions of precision (cf. Equation 3.25) and recall (cf. Equation 3.24) and the F_1 measure which assigns equal importance to both precision and recall.

Chapter 4

Modelling Approaches for Imbalanced Domains

4.1 Introduction

Imbalanced domains raise significant challenges when building predictive models. The scarce representation of the most important cases leads to models that tend to be more focused on the normal examples, neglecting the rare extreme events. As mentioned before, the problem of imbalanced distributions was initially addressed within a classification setting. Therefore, a large number of solutions was proposed specifically for classification tasks. These approaches aim to make the models focus on the less frequent and more important cases for the user. For instance, the model will be more focused in the rare cases if it is given the same number of rare and normal cases. Another example of a possible strategy is to modify the learning system internally so that it gives more attention to the rare examples.

All the strategies for handling imbalanced domains are traditionally separated in two groups named *internal* and *external* methods. The *internal* approaches aim at creating new algorithms or modify the ones already existing strengthening the learning process towards the least represented cases (cf. Figure 4.1). On the other hand, *external* approaches are usually connected to modifications on the data set previous to the learning process. These approaches try to manipulate the data, altering the existing data distribution, to get a more balanced sample and therefore reducing the effect of the imbalanced domain. We claim that *external* approaches should also include strategies which only make modifications on the predictions, i.e., methods that use the given data set and a standard learning algorithm and act only on the predictions by altering them to better correspond to the user preference bias. In summary, we consider that all the data set manipulations made previously to the learning process, and also the modifications applied after the standard learning algorithm to the predictions are examples of *external* approaches (cf. Figure 4.2).

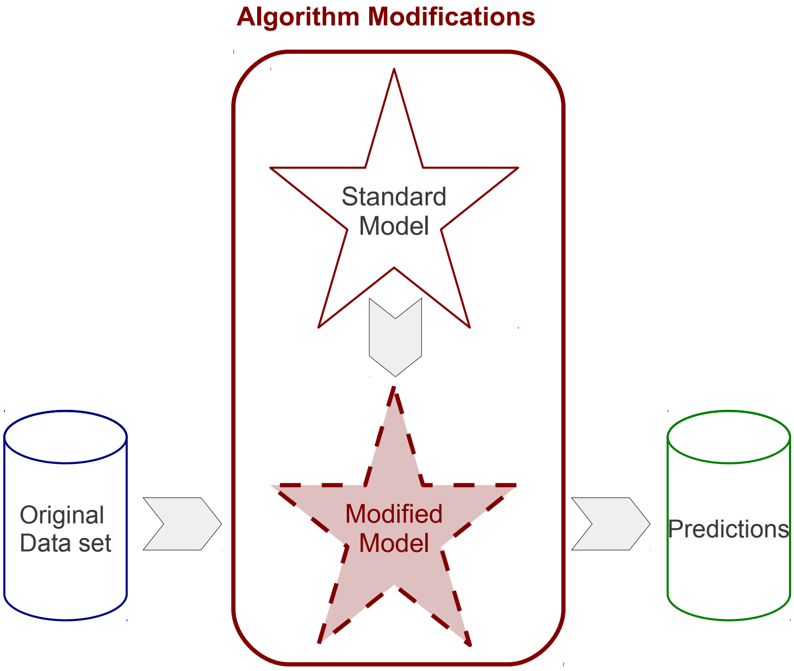


Figure 4.1: Internal approaches: Algorithm Modifications

In this context, we propose to cluster the existing approaches to learning under imbalanced data distributions in three different main groups:

External Approaches:

Data Pre-processing - Includes solutions that pre-process the given imbalanced data set, changing the data distribution so that the algorithm focus on the cases that are more relevant for the user ;

Prediction Post-processing - Approaches that use the original data set and an unchanged standard learning algorithm, only manipulating the models predictions to better adapt to the imbalanced problem;

Internal Approaches:

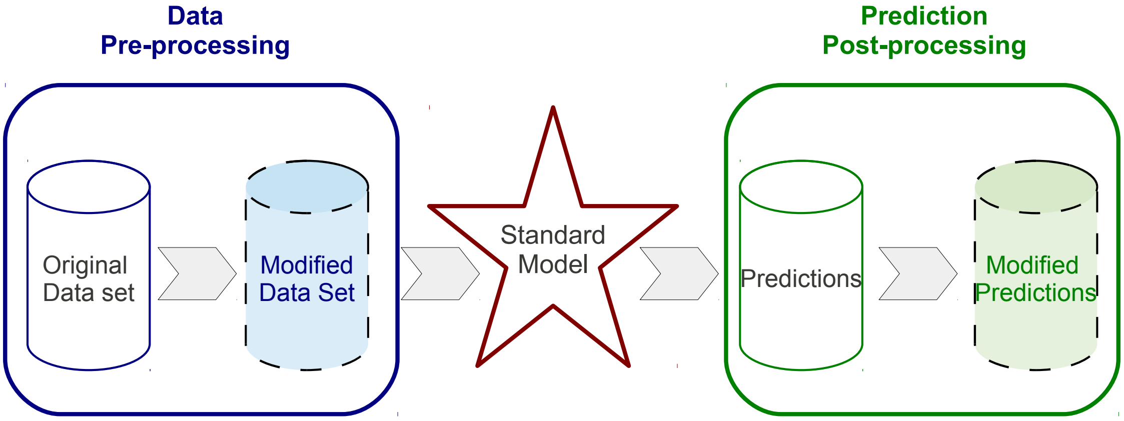


Figure 4.2: External approaches: Data Pre-processing and Prediction Post-processing

Algorithm Modifications - Comprises solutions which change the existing algorithms to provide a better fit to the imbalanced data .

For classification tasks several solutions exist following one of these alternatives or combinations of them into hybrid strategies. However, for regression tasks this issue is still under-explored, with only a few approaches included in the algorithm modifications and prediction post-processing strategies.

Each group of solutions has advantages and drawbacks which we will briefly describe next. The first group of data pre-processing methods has the following advantages: i) it can be applied to any existing learning tool; ii) the chosen models are biased to the goals of the user (because the data distribution was previously changed to match these goals), and thus it is expected that the models are more interpretable in terms of these goals. The main inconvenient of data pre-processing is that it might be difficult to relate the modifications in the data distribution with the target loss function, which may lead to worse results and models eventually not that comprehensible. The task of mapping the given data distribution with an optimal new distribution according to the user goals is not easy. As for the algorithm modifications group the following are important advantages: i) the user goals are incorporated directly into the models, or new models are constructed specially for the user goals; ii) it is expected that the models obtained this way are more comprehensible to the user. The main disadvantages of these approaches are: i) the user is restricted in his choice to the learning algorithms that were modified to be able to optimise his goals, or has to develop new algorithms for the task; ii) if the target loss function changes the model must be relearned, and moreover it may be necessary to introduce further modifications in the algorithm which may not be straightforward; iii) it requires a deep knowledge of the learning algorithms implementations. Finally, the last group of approaches presents the advantages: i) it is not necessary to be aware of the user preference bias at learning time; ii) the obtained model can in the future be applied to different deployment scenarios (i.e. different loss functions), without the need of re-learning the models or even keeping the training data available for this re-learning; iii) any standard learning tool can be used. However, this type of methods also have some drawbacks: i) the models do not reflect the user preferences; ii) the models interpretability is meaningless as they were obtained optimising a loss function that is not in accordance with the user preference bias.

In this thesis, we propose to address the problem of imbalanced domains for regression tasks through re-sampling strategies which are included on the data pre-processing group. These approaches have not yet been tried for regression.

4.2 Data Pre-processing Strategies

Pre-processing methods act on the given data set altering it so that it will be better adapted to the user preferences. Solutions at this level do not modify neither the algorithms nor the predictions made. Instead they include a pre-processing step, which modifies the data set distribution to force the algorithm to focus on the cases that are more relevant for the user.

As we have mentioned, several advantages justify the choice of these approaches. They allow the user to choose his preferred learning system without having to make any changes to it, and are methods usually quite simple and easy to use. A diverse set of data level approaches exist, each one with its particular advantages and drawbacks. For dealing with imbalanced domains at a pre-processing level we will consider three main solution types:

- **re-sampling:** change the data distribution of the data set forcing the learner to focus on the least represented examples;
- **active learning:** actively selects the best (more valuable) examples to learn leaving the ones with less information to improve the learner performance;
- **weighting the data space:** modify the training set distribution with regards

to misclassification costs, such that the changed distribution is biased towards the costly examples.

4.2.1 Re-sampling

Re-sampling approaches can be regarded as a pre-processing step whose goal is to modify the given data distribution to force the learner to focus on the least represented examples. In order to change data distribution, several techniques were proposed. Re-sampling strategies aim at altering the data distribution usually attempting to obtain a more balanced one. These strategies exist only for classification, and thus our descriptions will be focused on these tasks.

It was proved that applying a pre-processing step in order to obtain a more balanced class distribution is an effective solution to the imbalance problem (e.g. Estabrooks et al. (2004); Fernández et al. (2008); Batuwita and Palade (2010a); Fernández et al. (2010)). When compared to an imbalanced data set, a more balanced distribution of the data improves performance. Moreover, it has been shown that sampling is also an effective method for dealing with extreme imbalance (Seiffert et al., 2007).

However, changing the data distribution may not be as easy as expected. In fact, it may not be straightforward to decide what is the optimal distribution as it differs from one data set to another, which may lead to worse results . For classification tasks, it was proved that having a perfectly balanced distribution ($|D_N| = |D_R|$) does not always provides optimal results (e.g. Weiss and Provost (2003)). A study to evaluate the effect of the class distribution of examples on classification trees performance was conducted by Weiss and Provost (2003) and a budget-sensitive progressive sampling algorithm was proposed yielding a good (nearly-optimal) classification performance. A wrapper framework was also proposed by Chawla et al. (2005, 2008) that aims at discovering the right amount of re-sampling for a data set based on the optimisation of some evaluation functions.

For classification problems, changing the class distribution of the training data improves classifiers performance on an imbalanced context because it imposes non-uniform misclassification costs. This equivalence between the two concepts of altering the data distribution and the misclassification cost ratio is well-known and was first clarified by Breiman et al. (1984).

4.2.1.1 Random Under-sampling and Random Over-sampling

In order to better balance the data distribution, two simple strategies can be used: under-sampling and over-sampling, both with some variants. Random under-sampling removes data from the original data set, thus reducing the sample size. A random sample of the majority class examples is selected and then joined with the minority class examples to form the final training data set. Random over-sampling acts inversely by adding data from the minority class. A random sample of examples belonging to the minority class is selected and added to the training data set. This procedure increases the size of the training set, and balances the class distribution by introducing replicas of the minority class examples. Both for under- and over-sampling the amount applied varies according to the target class distribution and the data set.

Although simple, both under-sampling and over-sampling have known drawbacks (McCarthy et al., 2005). Under-sampling may discard potentially useful data by reducing the sample size, which can lead to worse performance. Over-sampling may increase the likelihood of overfitting, since it will produce ties in the sample, especially when the over-sampling rate increases (Chawla et al., 2002; Drummond et al., 2003). The introduction of replicated examples may decrease the classifier performance and also increase the computational effort due to an augmented sample size. Moreover, over-sampling does not introduce new data thus leaving the problem of lack of data (see Section 2.2) unsolved.

4.2.1.2 Distance Based Methods

An approach based on distance for performing under-sampling was presented by Chyi (2003). This approach computes distances among existing examples to select which majority class examples will be included in the training set. Four different methods for selecting samples are proposed: the *nearest*, the *farthest*, the *average nearest*, and the *average farthest* representing distances between the majority and minority classes.

The *nearest* method starts by calculating, for every minority class example, the distances between all majority class examples and the minority ones. Then selects the majority class examples having the smallest distances to each minority class examples. Similarly the *farthest* approach selects the majority class examples which have the farthest distances to each minority class examples. In both methods some of the majority class examples might be duplicated. The *average nearest* approach begins by calculating, for every majority class example, the average distance to all minority class examples. Then selects the majority class examples having the smallest average distances. Similarly to the *average nearest*, the *average farthest* method selects the majority class examples which have the farthest average distances with all the minority class examples. The four approaches have the disadvantage of being very time consuming and are, therefore, unsuitable for large data sets.

Another method proposed by Mani and Zhang (2003), uses the k nearest neighbour (k-NN) classifier to achieve under-sampling. For the under-sampling strategy four different methods are defined: NearMiss-1, NearMiss-2, NearMiss-3, and the most distant method. On NearMiss-1 method the majority examples selected have the smallest average distance to the three closest minority class examples. NearMiss-2 selects the majority examples whose average distance to the three farthest minority class examples is the smallest. NearMiss-3 aims to ensure that every minority example has in its neighbourhood some majority examples, and to do so, for each minority example selects a given number of the closest majority examples. Finally, the most distance method selects the majority class examples whose average distance to the three closest minority class examples is the largest. The experimental results showed a similar performance for random under-sampling and NearMiss-2, and a worse performance for the other proposed methods.

4.2.1.3 Data Cleaning Methods

Several data cleaning methods have been used with success to improve the performance of classifiers by removing the overlap introduced with sampling techniques. Data cleaning approaches can be applied as a focused under-sampling strategy only removing examples from the majority class with certain unwanted properties or withdraw examples from both classes which have certain defined undesirable characteristics.

One of those methods is based on the Tomek links (Tomek, 1976) notion which essentially consists of points that are each others closest neighbours, but do not share the same class label. More formally, a pair (x_i, x_j) is a Tomek link if x_i and x_j have different class labels and $\nexists\, x_k : d(x_i, x_k) < d(x_i, x_j) \vee d(x_j, x_k) < d(x_i, x_j)$. According to this definition, when two instances form a Tomek link one of two things can happen: one of the instances is noise, or both instances are near the border. Therefore removing all the Tomek links helps cleaning up unwanted overlapping between classes. Tomek links offer the possibility of being used as an under-sampling method or as a data cleaning method. If we only remove Tomek links examples belonging to the majority class we are applying an under-sampling strategy, if Tomek links examples of both classes are eliminated we are performing a data cleaning method (Batista et al., 2004).

The under-sampling strategy presented in Kubat and Matwin (1997) is based on Condensed Nearest Neighbour Rule (CNN) Hart (1968). The notion of CNN is used to find a subset of the given training data which is a consistent set of examples. A subset $\hat{S} \subseteq S$ is consistent with S if $\hat{S}$ correctly classifies the examples in S using a 1-nearest neighbour. The algorithm to create $\hat{S}$ starts by defining this subset as one randomly selected majority class example and all minority class examples. Then a 1-nearest neighbour classifier is trained in $\hat{S}$ and tested in S. All the misclassified examples from S are then integrated in $\hat{S}$. The goal is to keep the majority class examples that are near the decision border eliminating all the others.

Also in Kubat and Matwin (1997) another under-sampling strategy is proposed called One-Sided-Selection (OSS) which combines Tomek links and CNN. In this approach, Tomek links are firstly used as an under-sampling strategy removing only examples from the majority class, and afterwards CNN is applied also eliminating examples from the majority class but this time those who are distant from the borderline.

A similar procedure is presented in Batista et al. (2004) also involving CNN and Tomek links but applied in the reverse order of OSS. This choice is motivated by the computationally demanding task of finding Tomek links which would be performed on a smaller data set.

Another approach is proposed by Laurikkala (2001), called Neighbourhood Cleaning Rule (NCL), which depends on the concept of Wilsons Edited Nearest Neighbor Rule (ENN) (Wilson, 1972). For each example, ENN removes it if at least two of the three nearest neighbours have a different class label from its label. NCL is an under-sampling technique which modifies the ENN to increase the data cleaning. For each majority class example, if the three nearest neighbours classification contradicts the example original class, the example is discarded. As for each minority class example, if the three nearest neighbours misclassified the given example, then the neighbours are eliminated.

Recently Naganjaneyulu and Kuppa (2013) proposed a strategy called Class Imbalance Learning using Intelligent Under-Sampling (CILIUS). This algorithm acts by eliminating the weak or noisy examples which are related to specific features identified according to a well-established filter and intelligent technique named correlation-based feature subset (CFS) (Hall, 1999). The strong examples from the majority class and the minority class examples are then merged to form a new data set.

There are also approaches that integrate data cleaning techniques with other resampling approaches. These methods will be presented in Section 4.2.1.8 since they involve the combination of different strategies.

4.2.1.4 Cluster Based Methods

Recently Xuan et al. (2013) studied the effect of imbalanced data sets on clustering algorithms and showed that the class imbalance can seriously influence the performance and efficiency of the clustering algorithm. The higher the imbalance ratio of the data set, the higher the adverse effects on the clustering performance.

Despite these difficulties, clustering methods provide a great flexibility which makes them suitable for addressing simultaneously several problems. A good example of this is the cluster-based oversampling (CBO) algorithm proposed by Jo and Japkowicz (2004) for dealing with the *within-class* and the *between-class* imbalance problem. CBO consists of clustering the training data of each class separately with the k-means technique and then performing random over-sampling in each cluster. All the clusters of the majority class are over-sampled until they reach the same cardinality of the largest cluster of this class. Let m be the final size of D_N and $minclust$ be the number of clusters of the minority class. Each minority class cluster is random over-sampled until each one contains $\frac{m}{minclust}$ examples. After applying CBO both classes are balanced.

Yen and Lee (2006, 2009) presented a different approach called under-sampling based on clustering (SBC) which starts by clustering the training data into k clusters. Then, for each cluster, a number of majority class examples is selected being as larger as higher is the proportion of majority class examples in the cluster. These majority class examples are then combined with all the minority class examples to obtain a new training data set.

Other methods for under-sampling based on clustering and distances are presented in Yen and Lee (2006, 2009). These methods differ from SBC approach in the way majority class examples are selected in each cluster. In fact, these methods combine SBC algorithm with the notion of distance introduced by Mani and Zhang (2003) and previously explained in Section 4.2.1.2.

Three different cluster based approaches are presented in Cohen et al. (2006). The first uses clustering to substitute all majority class examples by prototypes generated. The second approach relies on the agglomerative Hierarchical Clustering (AHC) to over-sample the minority class. Finally, the third variant proposed involves the combination of AHC-based oversampling and K-means based under-sampling.

4.2.1.5 Synthesising New Data

Another approach for dealing with the imbalance problem as a pre-processing step, is the generation of new synthetic data. Several methods exist for building new minority class examples and therefore balance the data distribution. Synthesising new data has the following advantages (Chawla et al., 2002; Menardi and Torelli, 2010): i) reduce the risk of overfitting which is introduced when replicas of the examples are inserted in the training set; ii) improve the ability of generalisation which was compromised by the over-sampling methods.

One of the most famous methods is the Synthetic Minority Oversampling TEchnique - SMOTE (Chawla et al., 2002). This innovative and powerful method has shown success in several applications. SMOTE algorithm over-samples the minority class by generating new synthetic data. This technique is then combined with random under-sampling of the majority class. Artificial data is created using an interpolation strategy which introduces a new example along the line segment joining a seed example and one of its k minority class nearest neighbours. The number of minority class neighbours (k) is a parameter defined by the user. For each minority class example a certain number of examples is generated according to a predefined over-sampling percentage . For each minority (seed) example, a synthetic example is generated as follows:

1. randomly select one of its k nearest neighbours,
2. take the difference between the neighbour and the seed feature vectors,
3. multiply this difference by a random number ranging from 0 to 1,
4. add the result to the seed feature vector.

This results in the selection of a random point along the line segment between two specific features. The minority class label is assigned to the new example. Classifiers are then learned on the new data set with the majority class under-sampled and the minority class ”smoted”.

The Smote strategy was originally designed for data sets with all numeric features. A variant called Smote -NC was proposed to handle data sets with both numeric and nominal predictors. The SMOTE -NC strategy starts by computing the median of standard deviations of all numeric features for the minority class. For determining the nearest neighbours of a minority class example the Euclidean distance is used for numeric features and the median previously computed is included for penalising the nominal features with different values. The numeric features of the new synthetic case are determined with the same interpolation technique. Regarding the nominal features values of the synthetic example, the value occurring in the majority of the k-nearest neighbours is given.

SMOTE blindly generates synthetic minority class examples without considering the majority class and this may cause overgeneralization (Yen and Lee, 2006; Maciejewski and Stefanowski, 2011; Yen and Lee, 2009). This strategy may be specially problematic in the case of highly skewed class distributions where the minority class examples are very sparse thus resulting in a greater chance of class mixture. These issues motivated the appearance of approaches based on the SMOTE algorithm (Barua et al., 2012; Han et al., 2005; Bunkhumpornpat et al., 2009; Chawla et al., 2003; He et al., 2008; Maciejewski and Stefanowski, 2011; Ramentol et al., 2012b; Verbiest et al., 2012).

A different approach for generating synthetic data was proposed by Lee (1999). The main goal was to avoid overfitting to the training data and improve generalisation for the test data in skewed binary classification. The key idea was to over-sample the minority class by producing noisy replicates of the rare cases while keeping the majority class unchanged. The over-sampling was performed by adding some normal noise to the trained observations therefore creating new synthetic examples. The algorithm requires the user to set two parameters: $repl$ and σ_{noise}, the first one representing the number of noisy-replicates to produce for each minority class example, and the second one representing the introduction or not of noise. Let $D = \{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{n}$ be the training set where $\mathbf{x}_i$ is a p-dimensional feature vector and y_i is the binary response with $y_i = 1$ being the rare class. The new training set is generated in the following way:

1. replicate each example $\langle \mathbf{x}_i, 1 \rangle$ $repl$ times adding to the training set D the new generated examples $\{\langle \mathbf{x}_i + \epsilon_{ik}, 1 \rangle\}_{k=1}^{repl}$ with $\epsilon_{ik} \sim \mathbf{N}_p(0, \sigma_{noise}^2 \sum_p)$ where $\sum_p$ is the $p \times p$ diagonal matrix $diag\{s_1^2, \ldots, s_p^2\}$ and s_l^2 is the sample variance of the l-th feature variable over the training data;
2. let examples $\langle \mathbf{x}_i, 0 \rangle$ unchanged.

This simple strategy was tested with success, and a new version was developed in Lee (2000). This new approach generates, for a given data set, multiple versions of training sets with added noise. Then, an average of multiple model estimates is obtained. This method has shown success, improving the performance of several classifiers.

The effect of adding Gaussian Noise had already been addressed by An (1996). In his approach a new example is built for each existing training example. The synthetic examples are generated by adding a random vector following a Gaussian distribution with mean zero and a covariance matrix that takes into consideration the values of each of the original training examples, and the same class label as the original used example. This procedure maintains the ratio between the majority and the minority class and duplicates the training set.

Another framework, named ROSE (Random Over Sampling Examples), for dealing with the problem of imbalanced classification is presented by Menardi and Torelli (2010) and is based on a smoothed bootstrap re-sampling technique. ROSE generates a completely new and approximately balanced training set $D^* = \{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{m}$ from the original training set $D = \{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{n}$. The size m of the new training set is a parameter defined by the user. The new D^* includes only artificial examples which are built in the following way: one observation is draw from the training set by giving the same probability to both existing classes; the synthetic example is generated on the neighbourhood of the selected observation, with width determined by the smoothing matrix $\mathbf{H}$. This approach combines techniques of over-sampling and under-sampling generating an increased sample of data from the rare class and a possibly decreased sample from the majority class. The choice of $\mathbf{H}$ is critical once different choices of the smoothing matrices lead to larger or smaller neighbourhoods of the observations from which the synthetic examples are generated. The method proposed (Menardi and Torelli, 2010) considers Gaussian Kernels and minimises the AMISE (asymptotic mean integrated squared error) under the assumption that the true conditional densities underlying the data follow a Normal distribution.

ROSE procedure has shown excellent performance, in comparison to other similar methods, whether in real or simulated data. Simulations have shown that, in most cases, ROSE outperforms SMOTE and such improvement is mainly evident for extreme levels of imbalance and small sample sizes.

The Sanger Network Based Extended Over-Sampling Method (SNEOM) is a method proposed by Martínez-García et al. (2012). This approach is based on Sanger neural network and is an extended procedure because it allows to over-sample both minority and majority class. First, a dimensionality reduction is performed based on Sanger network and then, perturbations, such as Gaussian noise, are added to the data obtaining synthetic examples correlated to the original ones. The main advantage of this method is that over-sampling is performed on the transformed space of the input data thus being a method capable of dealing with high dimensional data sets.

Liu et al. (2007) proposed a method called Generative Oversampling for creating new data points by learning from available training data. Generative Oversampling is a method for generating synthetic examples based on an assumed probability distribution of the data whose parameters are learned from the training data.

4.2.1.6 Adaptive Synthetic Sampling

The generation of synthetic examples has several advantages, although some drawbacks have also been identified. Adaptive synthetic sampling methods have been proposed to overcome the potential problem of over generalisation mainly associated with SMOTE. This limitation is attributed to the way synthetic examples are generated: each original minority class example gives rise to the same number of synthetic examples and the neighbourhood of the minority class examples is not considered, thus increasing the occurrence of overlapping between classes (Stefanowski and Wilk, 2007; Bunkhumpornpat et al., 2009).

To cope with this problem several adaptations of SMOTE were proposed. These new approaches include: i) using standard SMOTE algorithm in the beginning and afterwards eliminating some of the synthetic examples generated by some chosen method; ii) using SMOTE for generating examples only in some specific locations or for generating different numbers of new cases for each minority class example; and iii) using clustering techniques or rough set theory in combination with SMOTE; and many other.

Borderline-SMOTE algorithm (Han et al., 2005) is one of the proposed SMOTE variants. Han et al. (2005) presents two new minority over-sampling methods (borderline-SMOTE1 and borderline-SMOTE2) in which only the minority class examples near the borderline are over-sampled. This is the key difference between Borderline-SMOTE and SMOTE: while SMOTE generates synthetic examples for each minority class example, Borderline-SMOTE (version 1 and 2) only generates synthetic instances for those minority examples ”closer” to the border. Both approaches have as main motivation the tendency to misclassify examples near the borderline . The minority instances are clustered into three different regions: noise, borderline, and safe. The definition of those regions is based on the number of majority instances n present on the example k nearest neighbours. The region is safe if $0 \leq n < \frac{k}{2}$, is considered borderline if $\frac{k}{2} \leq n < k$ and is labelled as noise if $n = k$. The strategy Borderline-SMOTE1 starts by calculating the k nearest neighbours for each rare class example and then the borderline examples are determined.The borderline examples are then over-sampled in a SMOTE-like fashion computing the m minority class nearest neighbours. Borderline-SMOTE2 strategy not only generates new synthetic examples from each example in the borderline using its positive nearest neighbours, but also does that from its negative nearest neighbours. On the latter version, the new synthetic examples generated are guaranteed to be closer to the borderline example than the negative neighbour considered. For the considered data sets, experiments showed that both Borderline methods used with C4.5 trees improved True Positive Rate and F-measure for the minority class over the original SMOTE and simple random over-sampling.

A different approach, Safe-Level-SMOTE, presented by Bunkhumpornpat et al. (2009), seeks for a careful over-sampling by only generating synthetic instances on a safe position. Safe-Level-SMOTE takes into account the presence of majority class instances before generating synthetic examples. A coefficient called *safe level* is calculated for each minority class example and that coefficient will determine whether the example is considered noise or is in a safe area. The *safe level* of a minority class example is the number of other minority class examples among its k nearest neighbours. A *safe level* equal or close to 0 means that the given example is interpreted as noise, if the *safe level* is closer to k, then the example is located in a safe region of the minority class. A seed example p belonging to the minority class is selected for over-sampling. Then, an example n is selected as being one of the k minority class nearest neighbours of p. For n and p the k nearest examples are found in the full training set and the respective *safe levels*, $sl(p)$ and $sl(n)$, are calculated. The *safe level ratio*, defined as $slr = \frac{sl(p)}{sl(n)}$ is calculated and one of five different cases can happen:

1. if $sl(p) = 0$ and $sl(n) = 0$, no example is generated because both p and n are considered noisy examples;

2. if $sl(p) > 0$ and $sl(n) = 0$, example n is considered as noise and an example is generated by simply duplicating p once the algorithm wants to avoid the noise instance n;

3. if $slr = 1$, both examples have a similar neighbourhood and the new synthetic example will be generated along the line joining them, in the same way as in the original SMOTE;

4. if $slr > 1$, example p is considered safer than n thus the synthetic example is generated closer to p;

5. if $slr < 1$, example n is considered safer and the synthetic example is generated closer to n.

Experiments showed that Safe-Level-SMOTE performance evaluated by precision and F-measure is better than that of SMOTE and Borderline-SMOTE for the considered data sets.

The previous approaches, Borderline-SMOTE and Safe-Level-SMOTE generate synthetic examples in different regions of the imbalanced data set. As we have mentioned, Borderline-SMOTE operates on the border of the minority class, while Safe-Level-SMOTE acts inside the minority class far from the border. Recently Bunkhumpornpat and Subpaiboonkit (2013) proposed a tool called Safe Level Graph whose goal is to guide the choice of the best technique to apply among the two just pointed. Safe Level Graph uses the frequency percentage of safe level values from all the positive examples to determine which method to apply. The safe level graph distribution is classified as skewed to the right or left and this determines the selection of one from the two methods.

ADASYN algorithm (He et al., 2008) uses a different method to adaptively create different amounts of synthetic data. This approach generates more synthetic examples for minority class instances that are harder to learn. The algorithm works in the following steps:

1. calculate the total number of synthetic examples, T, to be generated in order to obtain the desired balanced ratio between the two classes;

2. for each minority class example x_i, find the $k$nearest neighbours according to the euclidean distance and calculate $\Gamma_i = \frac{N_i/K}{Z}$, where N_i is the number of majority class examples on x_i k nearest neighbours, and Z is a normalization constant;

3. Γ_i will then be used to calculate the number g_i of synthetic examples to be generated for each minority instance x_i: $g_i = \Gamma_i \times T$;

4. generate synthetic examples accordingly to SMOTE algorithm.

The core idea of ADASYN is to automatically decide the number of synthetic examples that need to be generated for each minority instance by adaptively changing the weights of different minority examples to compensate for the imbalanced data distribution.

Another alternative approach is Modified Synthetic Minority Oversampling Technique (MSMOTE) proposed by Hu et al. (2009) which clusters the minority class examples into three groups, safe, border and latent noise based on the distance among all examples. MSMOTE selection of nearest neighbours depends on the group previously assigned to the instance. Thus, for safe instances, the algorithm randomly selects a data point from the k-nearest neighbours just like SMOTE; for border instances, it only selects the nearest neighbour; and for latent noise instances, it makes no selection.

Barua et al. (2012) presented MWMOTE which starts by identifying the hard-to-learn informative minority class examples and assigns them weights according to their Euclidean distance from the nearest majority class examples . It then generates the synthetic examples from the weighted informative minority class examples, by interpolation, using a clustering approach. This is done in such a way that all the generated examples lie inside some minority class cluster.

Batista et al. (2004) describe SMOTE+Tomek and SMOTE+ENN strategies, two new SMOTE based techniques. With the goal of creating better-defined class clusters, the first method applies Tomek links to the over-sampled training set as a data cleaning method. This strategy starts by over-sampling the data set applying SMOTE, then Tomek links are identified and removed producing well defined class clusters. In this case examples from both classes are removed. The second proposed method, SMOTE+ENN, is similar to SMOTE+Tomek links and also removes examples from both classes. ENN acts by removing examples that are misclassified by its three nearest neighbours and tends to withdraw more examples than Tomek links thus providing a more in depth data cleaning.

An improvement of SMOTE algorithm is proposed by Ramentol et al. (2012b), where the quality of the generated synthetic instances is monitored using fuzzy rough set theory. This approach called SMOTE-FRST, starts by applying SMOTE and then, iteratively, removes synthetic minority instances, as well as original majority instances, that have a small membership degree to the fuzzy positive region. Eliminated instances are regarded as noise and are filtered out from the training data. The process stops when the data set is balanced. A proposal also involving rough set theory, SMOTE-RSB, is presented by Ramentol et al. (2012a). SMOTE-RSB starts by generating synthetic examples with SMOTE algorithm and then applies a cleaning method based on rough set theory to include the original examples and the synthetic minority examples that belong to the lower approximation of their class in the final training set.

A prototype selection technique, Fuzzy Rough Imbalanced Prototype Selection (FRIPS), is presented by Verbiest et al. (2012). This approach aims to identify and clean noisy data before applying SMOTE, so that SMOTE can generate high quality artificial data. FRIPS deletes examples whose noise level (measured using fuzzy rough set theory) exceeds a certain threshold. This noise level threshold is determined using a wrapper approach that evaluates the training AUC of candidate subsets.

Another approach, FSMOTE, inspired on the theory of fractal interpolation was proposed by Zhang et al. (2011). Considering that all the minority examples obey the distribution of self-similarity and dilation symmetry in space, then the interpolated examples must also obey it. FSMOTE strategy generates examples which obey the spatial distribution of the original minority class examples with a deeper degree.

LN-SMOTE algorithm (Maciejewski and Stefanowski, 2011) focus on the local neighbourhood of the seed minority example, determining the k nearest neighbours in the training set also including the majority class ones. The idea is to avoid looking for minority class examples that are too distant. New synthetic examples are generated closer or further apart from the seed example depending on the local neighbourhood characteristics. More recently, García et al. (2012) presented three SMOTE based approaches for generating artificial minority instances that explore an alternative neighbourhood formulation named *surrounding neighbourhood*. These methods take into account both the proximity and the spatial distribution of the examples showing some practical advantages over the conventional neighbourhood that is simply based on the minimum distance.

Other approaches exist such as DBSMOTE algorithm (Bunkhumpornpat et al., 2012) which is based on DBSCAN clustering and SMOTE and LLE-SMOTE method (Wang et al., 2006) which uses a combination of the locally linear embedding algorithm (LLE) and SMOTE. Recently, LVQ-SMOTE (Nakamura et al., 2013) was proposed to tackle the difficulty of estimating proper borderlines between classes due to a huge feature space that is frequent in biomedical data. This method tries to generate synthetic examples to occupy more feature space than the existing SMOTE algorithms, and performs over-sampling using codebooks obtained by LVQ (Learning Vector Quantization).

4.2.1.7 Evolutionary Sampling

Evolutionary Algorithms (EA) are stochastic search methods that use mechanisms inspired by biological evolution named probabilistic operators such as mutation, selection and recombination. They rely on the concept of population of individuals representing candidate solutions to the optimization problem where the defined fitness function determines the quality of the solutions.

The EA have been used in several tasks with good results (Dehuri et al., 2008). For instance, these algorithms were applied in feature and instance selection with success (Whitley et al., 1997; García et al., 2008a). Only recently these methods were applied in imbalanced domains for classification tasks. In the context of imbalanced data sets, under-sampling can be regarded as a Prototype Selection (PS) procedure with the purpose of balancing the domain to achieve a better performance. This has motivated the use of EA as an under-sampling strategy for imbalanced domains.

García et al. (2006a) proposed a new evolutionary method for balancing the training set. The presented method uses a new fitness function designed for performing a prototype selection process with the goal of balancing data, improving the generalisation capability and reducing the training data. Some proposals have also emerged in the area of heuristics and metrics for improving several genetic programming classifiers performance in imbalanced domains (Doucette and Heywood, 2008).

Evolutionary Under-Sampling (EUS) is an approach proposed by García and Herrera (2009) which uses EA for under-sampling imbalanced domains. In order to do so, several data subsets are randomly under-sampled, being then evolved until the currently best under-sampled data set cannot be further improved (in terms of the fitness function). Eight different EUS methods are presented and categorised into a taxonomy depending on their objective, scheme of selection and metrics of performance employed.

Regarding the objective, methods that aim for an optimal data balancing are named Evolutionary Balancing Under-Sampling (EBUS) while those aiming for an optimal power of classification without taking into account data balancing are called Evolutionary Under-Sampling guided by Classification Measures (EUSCM). Another distinction is made regarding the instance selection procedure which can be a Global Selection (removals from the minority class are allowed) or a Majority Selection (minority class instances removal is not allowed). Finally, methods are distinguished based on the metric used in the fitness function.

A solution named Evolutionary Sampling is proposed by Drown et al. (2009) and applied to the specific context of improving software quality modelling for high-assurance systems. The proposed approach uses Genetic Algorithms (GA) for under-sampling the majority class and a fitness function that optimises two commonly used performance metrics: AUC and G-Mean.

However, EA have been used for more than under-sampling. In fact, in the work of Maheshwari et al. (2011) a combined strategy of GA and clustering techniques is presented. Different GA operators are used for over-sampling to enlarge the ratio of positive examples and then clustering is performed on the over-sampled training set as a data cleaning method for both classes, removing the redundant or noisy examples. Following a reverse path, in the proposal of Yong (2012), the K-means algorithm is first applied on the minority class examples and then a genetic algorithm is used.

Also, the study of Derrac et al. (2012) presents EGIS-CHC, an evolutionary model to improve imbalanced classification based on nested generalized example that accomplishes learning by storing objects in Euclidean n-space. New examples are classified by computing their distance to the nearest generalized exemplar. The proposed strategy performs an optimized selection of the most suitable generalized exemplars based on evolutionary algorithms and is also combined with SMOTE pre-processing yielding to simpler models.

4.2.1.8 Combining Re-sampling Strategies and Other Strategies

Sometimes, different types of the previously presented strategies are combined and/or altered to improve the performance of learning systems under imbalanced domains. Those associations and modifications will be explored on this section.

The SPIDER algorithm (Selective Preprocessing of Imbalanced Data) proposed by Stefanowski and Wilk (2008) combines local over-sampling of the minority class with filtering difficult examples from the majority class. The first step is to identify which instances are flagged as *noisy* and which are considered *safe*. Examples that are correctly classified by its k nearest neighbours are *safe*, and the others are *noisy*. The second step depends on a parameter which can be set as: *weak*, *relabel*, or *strong*. If *weak* is chosen, the minority class instances that were misclassified are over-sampled by introducing copies of those instances. For the *relabel* option, an extension of the previous option is done adding a modification to the majority class instances . Finally, for the *strong* option, the minority class instances are strongly amplified.

After carrying out these operations, the remaining noisy examples from the majority class are removed from the data set. Classification performance of SPIDER approach is slightly better or comparable to SMOTE thus being a possible alternative to this one.

SPIDER algorithm first identifies the nature of the examples and then simultaneously processes the majority and minority class. Nevertheless this processing can result in too extensive modifications in some regions of the majority class and may deteriorate specificity. This drawback was addressed by Napierała et al. (2010) with SPIDER2 method. This algorithm consists of two phases for pre-processing examples of the majority class and minority class respectively.

Another technique called MUTE is presented in Bunkhumpornpat et al. (2011) for addressing the problem of an enlarged data set originated by over-sampling strategies. When over-sampling is used to adjust the class distribution, the computation of generating a classifier is highly affected due to an increased data set size. MUTE is a new simple and effective under-sampling strategy with the purpose of discarding the noise majority instances which overlap with minority instances. The removal of the majority instances is based on their safe levels which in turn relies on the Safe-Level-SMOTE concept. MUTE withdraws from the original data set all the majority instances that are considered noise, returning a reduced data set. MUTE has the advantage of reducing the time spent constructing a classifier due to a reduction of the data set size. Results also show that MUTE improves the F_β comparing to SMOTE, Borderline-SMOTE and Safe-Level-SMOTE.

In Songwattanasiri and Sinapiromsaran (2010)a new technique called Synthetic Minority Over-Sampling and Under-sampling Technique (SMOUTE) is presented which combines SMOTE over-sampling with under-sampling by reduction around centroids. The main idea of SMOUTE algorithm, is to avoid synthesize a large number of minority class instances while balancing both classes.

Vasu and Ravi (2011) propose an approach for performing informed under-sampling which tries to eliminate the noisy and redundant examples from the majority class. The method first applies k-reverse nearest neighbour (k-RNN) for detecting and removing noise from the majority class and then uses the K-means clustering algorithm for redundancy removing. This method was tested with success on fraud detection and credit churn modelling problems. Yang and Gao (2012) presents an active under-sampling approach. This method, instead of discarding the majority class examples randomly, actively selects the examples of the majority class which are near the decision boundary, maintaining at the same time the original density distribution. The idea is to put apart the abundant majority class examples based on the density data distribution.

An hybrid method is proposed in Li et al. (2008) for dealing in particular with the improvement of SVMs performance in an imbalanced context. This approach is motivated by the need to overcome some detected flaw of the traditional re-sampling methods and some data confusion. A variable self-organizing map (SOM) clustering is used for re-sampling the data set. Then the training set is pruned by means of k-NN rule to solve the problem of data confusion. The two steps improve the generalization ability of SVM under imbalanced domains.

4.2.2 Active Learning

Active learning is a semi-supervised learning strategy in which the learning algorithm is able to interactively obtain information. This strategy actively selects the best, i.e. the most informative, examples to learn. The more valuable examples are selected and those with less information are abandoned, with the goal of improving the learner performance. Active learning techniques are traditionally used to solve problems related to unlabelled training data.

Nonetheless, recently, several approaches for imbalanced data sets based on active learning have been proposed (Ertekin et al., 2007b,a; Zhu and Hovy, 2007; Ertekin, 2013).

Ertekin et al. (2007b,a) proposed an active learning method based on SVMs. This approach avoids searching the entire training data space, and can effectively select informative instances from a random set of training populations. This way, when dealing with large data sets, the computational cost is significantly reduced. The selection strategy, named SVM based active learning, is based on the fact that, for SVMs, the most informative instance is believed to be the closest instance to the hyperplane.

Active learning was also used in the context of class imbalance problems in word sense disambiguation applications (Zhu and Hovy, 2007). Strategies as max-confidence and min-error were investigated as the stopping criteria for the proposed active learning methods.

An active learning method for imbalance data using the Localized Generalization Error

Model (L-GEM) of radial basis function neural network (RBFNN) was presented by Hu (2012).

More recent developments try to combine active learning with other techniques (Ertekin, 2013; Mi, 2013) to further improve learners performance. Ertekin (2013) presents a novel adaptive over-sampling algorithm, VIRTUAL, that combines the benefits of over-sampling and active learning. VIRTUAL generates synthetic examples for the minority class during the training process. Therefore, the need for an extra pre-processing stage is discarded. In the context of learning with SVMs, VIRTUAL outperforms competitive over-sampling techniques both in terms of generalisation performance and computational complexity.

In the work of Mi (2013) a new method is developed by introducing SVM into the learning framework of SMOTE for class imbalance learning. The proposed method uses active learning SMOTE to classify the imbalanced data. In this study, the SMOTE method is adapted for advancing the classification of imbalanced data.

4.2.3 Weighting the Data Space

The strategy of weighting the data space is a way of implementing cost-sensitive learning. In fact, misclassification costs are applied to the given data set with the goal of selecting the best training distribution. Essentially, this method is based on the fact that changing the original data distribution to another, multiplying each example by a factor that is proportional to the importance (relative cost), makes any standard learner accomplish expected cost minimisation on the original distribution. Although it is a simple technique and easy to apply some drawbacks exist. There is a risk of model overfitting and is also possible that the real cost values are unavailable which can introduce an extra learning cost for the need of exploring effective cost setups.

This approach has a strong theoretical foundation, building on the *Translation Theorem* derived in Zadrozny et al. (2003). So, to obtain a modified distribution biased towards the costly classes, the training set distribution is modified with regards to misclassification costs. Let us consider a *normal space* without the cost item with domain $X \times Y$, and a *cost space* with domain $X \times Y \times C$, where X is the input space, Y is the output space and C being the cost associated with mislabelling an example.

If we draw examples from a distribution D in the *cost space*, then we can have another distribution $\hat{D}$ in the *normal space* such that

$$\hat{D}(X,Y) \equiv \frac{C}{E_{X,Y,C\sim D}[C]} D(X,Y,C) \tag{4.1}$$

where $E_{X,Y,C\sim D}[C]$ is the expectation of cost values.

According to the *Translation Theorem*, those optimal error rate classifiers for $\hat{D}$ will be optimal cost minimizers for D. Thus, when we update sample weights integrating the cost items, choosing a hypothesis to minimize the rate of errors under $\hat{D}$ is equivalent to choosing the hypothesis to minimize the expected cost under D.

Zadrozny et al. (2003) presents two different ways of accomplishing this conversion: in a transparent box manner by feeding the weights to the classification algorithm or in a black box manner by carefully sub-sampling accordingly to the same weights. However, the first approach cannot be applied to an arbitrary learner, and the second one results in severe overfitting if re-sampling with replacement is used. Thus, to overcome the drawbacks of the later approach Zadrozny et al. (2003) presented a method called *cost-proportionate rejection sampling* which accepts each example in the input sample with probability proportional to its associated weight.

4.3 Modifications on the Algorithms

The approaches at this level consist of solutions for modifying the existing algorithms to provide a better fit to the imbalanced data. The task of developing a solution based on algorithm modifications is not an easy one. It requires a deep knowledge of both the learning algorithm and the target domain. To perform a modification on a selected algorithm it is essential to understand why it fails when the distribution is skewed. Also, some of the adaptations assume that a cost-matrix is known for different error types, which is frequently not the case. On the other hand, these methods have the advantage of being very effective in the context for which they were though for.

For dealing with imbalanced domains at the algorithm level we will consider three main solution types:

- **recognition-based methods:** a model is obtained with only examples of the target class in the absence of the counter examples. This approach does not try to partition the hypothesis space with boundaries that separate positive and negative examples, but it attempts to make boundaries which surround the target concept;

- **cost-sensitive algorithms:** costs are incorporated directly in the algorithm, adapting the standard learning method and making it cost-sensitive.

- **development of new algorithms:** new algorithms are developed to specifically deal with this problem.

4.3.1 Recognition-based Methods

Recognition-based methods as one-class learning have also been applied in imbalanced domains with promising results (Chawla et al., 2004). In this type of approach, and contrary to discrimination-based inductive learning, the model is obtained using only examples of the target class, and no counter examples are included. This lack of examples from the other class(es) is the key distinguishing feature between recognition-based and discrimination-based learning. The use of these methods was motivated by many real world situations where it is only possible to have data from one class (the target class) being data from other classes (the outlier classes) very difficult or even impossible to obtain (Bellinger et al., 2012).

One-class learning does not try to partition the hypothesis space with boundaries that separate positive and negative examples. The effort is directed towards setting up boundaries which surround the target concept. Essentially, the goal of this method is to measure the amount of similarity between an object and the target class, and classification is accomplished by imposing a threshold on the similarity measure. The major drawback of one-class learning methods is the need for tuning the similarity threshold. Choosing a narrow threshold means that positive data will be discarded, while a wide threshold will include a considerable number of negative examples. Therefore, establishing an efficient threshold is vital with this method. Also, some learners actually need examples from more than one class and are unable to adapt to this method. Despite all these possible disadvantages, recognition-basedlearning algorithms have been proved to provide good prediction performances in most domains.

Developments made in one-class learning include one-class SVMs (e.g. Schölkopf et al. (2001); Manevitz and Yousef (2002); Raskutti and Kowalczyk (2004); Zhuang and Dai (2006); Lee and Cho (2006)) and the use of an autoencoder (or autoassociator) (e.g. Japkowicz et al. (1995, 2000); Japkowicz (2001b); Manevitz and Yousef (2007)).

The one-class SVM was first proposed by Schölkopf et al. (2001) to estimate the probability density function where the data set is drawn from. This method assumes that the origin in the kernel space is the second class, and, subsequently, learns a boundary that separates the target class from the origin. In Manevitz and Yousef (2002) an extension of this approach is proposed. This new version called "outlier" methodology assumes not only that the origin is in the negative class but also includes the points which are "close enough" to the origin. This method uses a threshold which is empirically determined. However, in addiction to the difficulty in determining the threshold, we should also consider the issue of choosing the SVM parameters and the SVM kernel as reported by Manevitz and Yousef (2002). Apart from the difficulties, one-class SVM has showed very good performance particularly for small or extremely imbalanced data sets (Manevitz and Yousef, 2002; Raskutti and Kowalczyk, 2004).

Another recognition-based method is the autoencoder (Hinton, 1989) which can be thought of as a compression neural network, where the goal is to try to recreate the input at the output, i.e., is a neural network which maps the inputs to output nodes, through a narrow hidden layer, attempting to reconstruct the input. The narrow hidden layer forces the compression of redundancies in the input while retaining and differentiating non-redundant information. The network is trained to learn the identity function on a training set consisting of positive examples only. The autoencoder should then be able to adequately reconstruct subsequent positive instances, but should perform poorly on the task of reconstructing subsequent negative instances. Therefore, positive and negative instances are identified by assessing how well such instances are reconstructed by the autoencoder. Under certain conditions such as multimodal domains, the one-class learning may be superior to the discrimination-based approaches (Japkowicz, 2001b) , being an useful method for extremely imbalanced data sets composed of a high dimensional noisy feature space (Raskutti and Kowalczyk, 2004).

A novelty detection approach based on an autoencoder was studied in Japkowicz et al. (1995). It is suggested that novelty detection methods are more useful for extremely imbalanced data sets, while for moderate imbalanced data sets the regular discrimination-based classifiers bring more benefits (Lee and Cho, 2006) .

A more recent study Bellinger et al. (2012) investigated the performance variations of binary and one-class classifiers for different levels of imbalance. The results on both artificial and real world data sets showed that as the level of imbalance increased, the performance of binary classifiers decreased, whereas the performance of one-class classifiers stayed relatively stable. This study confirms the conclusions of previous ones, pointing that when the level of imbalance is extreme, recognition-based methods may provide a better performance.

4.3.2 Cost-sensitive Algorithms

Some algorithms can directly incorporate costs as a way for improving the performance in imbalanced domains. A standard learner can be adapted to be cost-sensitive so that it take into consideration costs. In this case, the goal of the prediction task is to minimize the total cost, knowing that misclassified examples may have different costs. A fundamental concept in cost-sensitive learning is the notion of a cost-matrix which expresses the numeric penalty for different types of errors. For classification tasks, let $C(i, j)$ be the cost of predicting an example from class i as a class j. Then, for binary classification $C(min, maj)$ is the cost of misclassifying a minority class example as a majority instance, and $C(maj, min)$ is the cost of the contrary. In an imbalanced context, the cost of misclassifying a minority class example is superior than the cost of misclassifying a majority class example, i.e. $C(min, maj) > C(maj, min)$ and usually there is no cost associated with making a correct prediction, i.e. $C(min, min) = C(maj, maj) = 0$.

Making decision trees cost-sensitive can be accomplished in three different ways: the decision threshold can be integrated with costs; the splitting criterion at each node can consider costs; and, finally, the tree pruning schemes can incorporate costs. Maloof (2003) uses the ROC evaluation procedure for determining the optimal decision threshold which is then used in the decision tree. Works as Ling et al. (2004); Elkan (2001); Drummond and Holte (2000) address the introduction of cost sensitivity in the split criterion of decision trees. Although pruning is beneficial for decision trees by allowing to improve generalization, when applied on imbalanced data sets this procedure has an undesirable behaviour tending to remove leaves describing the minority concept. Also, leaving the decision trees unpruned does not improve the performance in such domains. Thus, works such as the Laplace smoothing method of the probability estimate and the Laplace pruning technique (Elkan, 2001) try to improve the class probability estimate in each node so that pruning can be applied with a positive effect.

The Iterative Bayes method that modifies Naive Bayes to accommodate asymmetric cost structures was proposed by Gama (2003).

Some research has also been conducted on support vector machines in order to make them cost-sensitive. The most straightforward technique for integrating costs into SVM modelling, implemented in LIBSVM (Chang and Lin, 2011), is to assign a larger penalty value to false negatives than false positives (Veropoulos et al., 1999; Akbani et al., 2004). Still, several other proposals were made for making SVMs cost-sensitive.

For instance, in Tang et al. (2009) SVM-WEIGHT is presented, the work of Yuanhong et al. (2009) proposes a cost-sensitive SVM approach based on weighted attribute, and in Hwang et al. (2011) the approach of SVMs with asymmetric costs was reported to be efficient. In Fumera and Roli (2002) is proposed an extension of SVMs that directly embeds reject option. Weiguo et al. (2012) proposes a new method based on SVM-KM algorithm (Barros de Almeida et al., 2000). SVM-KM model can speed SVM training by eliminating non support vectors using the k-means clustering algorithm. The improved SVM-KM model presented, assigns different error costs to different classes, so that the learner can better deal with the imbalance problem.

Regarding neural networks, the possibility of making them cost-sensitive has also been considered (Zhou and Liu, 2006; Alejo et al., 2007; Oh, 2011). A Cost-Sensitive Multilayer Perceptron (CSMLP) algorithm is proposed in Castro and de Pádua Braga (2013) for asymmetrical learning of MLPs via a modified (backpropagation) weight update rule. In Cao et al. (2013) a framework based on Particle Swarm Optimization (PSO) for improving the performance of cost-sensitive neural networks is presented. PSO is used for simultaneously optimize misclassification cost, feature subset and intrinsic structure parameters. Alejo et al. (2007) proposes two strategies for dealing with imbalanced domains using RBF neural networks. The first method includes a cost function in the training phase to compensate the imbalance in the training set. However, adding a cost function to the training phase causes changes in data probability distribution. This has motivated a second strategy to reduce the impact of the cost function in the data probability distribution. Thus, the second method gradually modifies the cost function until it does not have any influence.

Ensembles have also been considered in the cost-sensitive framework. Several ensemble methods have been successfully adapted to include costs during the learning phase. However, boosting was the most extensively explored.

AdaBoost is the most representative algorithm of boosting family. When the class distribution is imbalanced AdaBoost biases the learning (through the weights) towards the majority class, since it contributes more to the overall accuracy. This has lead to several proposals which modify AdaBoost weight update process by incorporating cost items so that examples from different classes are treated unequally. Important proposals in this context are: AdaCost (Fan et al., 1999), CSB1 and CSB2 (Ting, 2000), RareBoost (Joshi et al., 2001), AdaC1, AdaC2 and AdaC3 (Sun et al., 2007), and BABoost (Song et al., 2009). All of them modify the AdaBoost algorithm by introducing costs in the weights update formula used. These proposals differ in how they modify the weight update rule.

Random Forests have also been adapted to better cope with unbalanced data sets undergoing a cost-sensitive transformation. In Chen et al. (2004) is proposed a method called Weighted Random Forest (WRF) for dealing with highly-skewed class distributions based on the Random Forest algorithm. WRF strategy uses the idea of cost-sensitive learning. By assigning a higher misclassification cost to the minority class, WRF improves classification performance of the minority class and also reduces the total cost. For a more complete review on ensembles for the class imbalance problem see Galar et al. (2012).

Incorporating costs on the algorithms has been applied successfully for several classifiers. However, some disadvantages exist and should be mentioned such as: an often unavailable cost-matrix, a need of a deep knowledge of the selected learner to accomplish a good incorporation of costs and the poor portability of the method which contrast with pre-processing approaches.

4.3.3 Development of New Algorithms

In this section we describe some of the existing work regarding the development of new algorithms. The main goal is to adapt existing learners to better focus on the rare examples. Modifications on several learners were proposed and also combinations of algorithms producing a new strategy have been presented. As in other approaches, this type of strategies was mainly applied to classification tasks.

Regarding Support Vector Machines (SVM), some proposals try to bias the algorithm so that the hyper-plane is further away from the positive class as the skew associated with imbalanced data sets pushes the hyper-plane closer to the positive class. Wu and Chang (2003) have accomplished this biasing with an algorithm that changes the kernel function.

Another approach also related with the introduction of modifications into SVM learners is called z-SVM (Imam et al., 2006) and aims at obtaining a good margin between the decision boundary and each of the classes, correcting the skew of the learned SVM model automatically, irrespectively of the choice of learning parameters and without multiple SVM training.

Tang and Zhang (2006) proposed the Granular Support Vector Machines - Repetitive Undersampling algorithm (GSVM-RU). This algorithm integrates SVM learning with undersampling techniques and is based on the notion of Granular Support Vector

Machines (GSVMs). GSVMs present the advantages of: improving the computational efficiency of SVMs through the use of parallel computing, and analysing the inherent data distribution by observing the trade-offs between the local significance of a subset of data and its global correlation. The GSVM-RU approach builds on an iterative learning procedure which uses the SVMs for under-sampling.

Fuzzy Support Vector Machines for Class Imbalance Learning (FSVM-CIL)was a method proposed by Batuwita and Palade (2010b). This algorithm is based on an SVM variant for handling the problem of outliers and noise called FSVM and improves it for also dealing with imbalanced data sets.

Potential Support Vector Machine (P-SVM) differs from standard SVM learners by defining a new objective function and constraints. Although offering many advantages, this method poses difficulties when learning from imbalanced domains since it uses the same penalty for positive and negative slack variables. In this context, an improved P-SVM algorithm (Li et al., 2009) was proposed to better cope with imbalanced data sets. This new approach introduces flexibility in the adjustment of penalty parameters of the positive and negative slack variables.

Also k-NN learners were adapted to better deal with the imbalance problem. Barandela et al. (2003) presents a weighted distance function to be used in the classification phase of k-NN without changing the class distribution. This method assigns different weights to the respective classes and not to the individual prototypes. Since more weight is given to the majority class, the distance to minority class examples becomes much lower than the distance to examples from the majority class. This biases the learner to find their nearest neighbour among examples of the minority class.

In Huang et al. (2004) is presented a new approach named Biased Minimax Probability Machine (BMPM) to address the imbalance problem which is based on extending the Minimax Probability Machine (MPM) algorithm (Lanckriet et al., 2003). The proposed BMPM method uses the reliable mean and covariance matrices of the majority and minority classes to derive the decision hyper-plane.

A new decision tree algorithm, Class Confidence Proportion Decision Tree (CCPDT) is proposed in Liu et al. (2010). CCPDT is robust and insensitive to class distribution and generates rules which are statistically significant. The algorithm adopts a new proposed measure, called Class Confidence Proportion, which forms the basis of CCPDT and defines a new approach to prune branches of the tree which are not statistically significant.

Hellinger distance was introduced as a decision tree splitting criterion to build Hellinger Distance Decision Trees (HDDT) (Cieslak and Chawla, 2008). This proposal was shown to be insensitive towards class distribution skewness. More recently Cieslak et al. (2012) recommended the use of bagged HDDTs as the preferred method for dealing with imbalanced data sets when using decision trees. The proposal of using Hellinger trees with bagging is mentioned to be sufficient under imbalanced domains and the authors stress that no sampling methods are needed.

Other strategies were proposed which involve the combination of algorithms. An example is the proposal of Phua et al. (2004) were stacking and boosting are used together. Stacking is a technique similar to boosting involving the training of a model by combining the predictions of several other learners. Instead of using weights, as boosting does, a new learner is trained with the outputs of the models already trained. In Phua et al. (2004) this approach is combined with bagging to identify the best mix of classifiers. For an insurance fraud detection domain, this approach achieved the best cost-savings.

Rodriguez et al. (2012) propose the combination of Disturbing Neighbours ensemble with bagging using three types of trees as base classifiers: conventional decision trees (C4.5), Hellinger Distance Decision Trees (HDDT) and model trees (M5P).

In Wu and Chang (2005) the Kernel Boundary Alignment algorithm (KBA) is proposed. This method adjusts the boundary towards the majority class by modifying the kernel matrix generated by a kernel function according to the imbalanced data distribution.

An ensemble method for learning over multi-class imbalanced data sets, named ensemble Knowledge for Imbalance Sample Sets (eKISS), was proposed in Tan et al. (2003). This proposal was specifically designed to increase classifiers sensitivity without losing the corresponding specificity and was applied for multi-class protein fold domain. The eKISS algorithm combines the rules of the base classifiers to generate new classifiers for final decision making. In this study, the PART rule-based machine learning technique was used to generate the base classifiers for the ensemble learning system. This method was also successfully extended for being able to learn over multiple data sources.

Recently, more sophisticated approaches were proposed as the Dynamic Classifier Ensemble method for Imbalanced Data (DCEID) presented by Xiao et al. (2012). DCEID combines ensemble learning with cost-sensitive learning and is able, for each test instance, to adaptively select the more appropriate from the two kinds of dynamic ensemble approach: Dynamic Classifier Selection (DCS) and Dynamic Ensemble Selection (DES). DCS and DES are two commonly used strategies for dynamic classifier ensemble. The first selects a single best classifier for each test sample while the second one selects an optimal classifier ensemble for each test sample. The DCEID proposal fuses DCS and DES strategies and constructs a new cost-sensitive selection criteria respectively for DCS and DES to specifically address the imbalanced data problem.

For regression tasks only one approach exists that addresses the problem of imbalanced domains through the development of new algorithms. This approach is called utility-based Rules (ubaRules) and was proposed by Ribeiro (2011). ubaRules is an utility-based regression rule ensemble system designed for obtaining models biased according to a specific utility-based metric. The system main goal is to obtain accurate and interpretable predictions in the context of regression problems with non-uniform utility. 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.

4.4 Post-processing the Predictions

For dealing with imbalanced domains at the post-processing level we will consider two main solution types:

- **threshold method:** each prediction is associated with a score that represents the degree to which an example is a member of a class; such score can be transformed in a ranking that can be used to produce several models, by varying the threshold of an example pertaining to a class;
- **cost-sensitive post-processing:** associates costs to prediction errors and minimizes the expected cost.

4.4.1 Threshold Method

Some classifiers are named soft classifiers because they yield a score that represents the degree to which an example is a member of a class. This score can, in fact, be used as a threshold to generate other classifiers. This task can be accomplished by varying the threshold of an example belonging to a class (Weiss, 2004). A study of this method (Maloof, 2003) concluded that the operations of moving the decision threshold, applying a sampling strategy, and adjusting the cost matrix produce classifiers with the same performance.

4.4.2 Cost-sensitive Post-processing

Several methods exist, although mainly for classification tasks, which use a standard learning algorithm and change only the predictions in order to make the model cost-sensitive. Even though these methods have not been applied in imbalanced domains specifically, we consider them as a viable option for this problem.

Domingos (1999) presented Metacost, a method for making an arbitrary classifier cost-sensitive by wrapping a cost-minimizing procedure around it. Metacost treats the classifier as a black box and the user is not required to have any knowledge of classifiers functioning neither it is necessary to change them. Metacost relabels training examples with their estimated minimal-cost classes, and applies the learner to the new training set. Essentially, Metacost procedure takes the chosen classifier and begins by learning an internal cost-insensitive model. Then, it uses a variant of bagging for estimating each class probability for each example and training examples are relabelled with the estimated optimal class. Finally the classifier is reapplied to the modified training set.

For regression problems, introducing costs at a post-processing level, has only recently been proposed. It is an issue still under-explored with few limited solutions. In Bansal et al. (2008) was proposed an algorithm which tunes the outputs of a trained regression model reducing its average misprediction cost (a new metric also proposed in this work). This post-processing method is able to deal with any convex cost functions without modifying the underlying regression algorithm. However, this method is rather restrictive since it only adjusts the predictions of a regular regression model by a certain constant amount. As a consequence of this disadvantage, the method proposed by Bansal et al. (2008) was extended in the work of Zhao et al. (2011). The latter, although following the same guidelines allows for the regular regression model to be adjusted with a polynomial function.

A proposal for addressing regression tasks named reframing (Hernández-Orallo, 2012) was recently presented. This approach tackles cost-sensitive problems in regression by the reuse (and not re-training) of general regression models acting as a post-processing technique. Reframing can be defined as the process of applying a previously built model to a new operating context by the proper transformation of inputs, outputs and patterns.

Although reframing was not developed specifically for imbalanced domains, it can be regarded as a method for incorporating costs at a prediction level, being a possible alternative for dealing with regression tasks under imbalanced domains. The reframing method essentially consists of two steps:

- the conversion of any traditional one-parameter crisp regression model into a two-parameter soft regression model, seen as a normal conditional density estimator (NCDE), by the use of enrichment methods;
- the reframing of an enriched soft regression model to new contexts by an instance-dependent optimisation of the expected loss derived from the conditional normal distribution.

Several enrichment methods are proposed to perform the conversion of a crisp regression model into a soft regression model by just comparing the output value y with the estimated output value $\hat{y}$.

4.5 Hybrid Approaches

In recent years an increasingly diverse range of approaches has been explored for classification problems. Important contributions to deal with the problem of imbalanced domains were made from all the types of strategies. In this context, a question that naturally arises is related to the combination of strategies of different kinds of approaches, i.e., hybrid methods. Regarding this issue several attempts were made and are addressed over the next sections. These methods essentially try to capture the best of two selected strategies of different types combining them into one. Hybrid methods can be cluster into: combining algorithms predictions, re-sampling integrated with algorithm modifications, and other hybrid strategies.

4.5.1 Combination of Algorithms Predictions

One of the first works for combining algorithms with the goal of improving performance in imbalanced domains was presented by Chan and Stolfo (1998). The proposed method starts with preliminary experiments to identify a good class distribution.

Afterwards, multiple training sets are generated with the previously determined target class distribution. It is ensured that no data is wasted by forcing each majority class example to be included in at least one of the training sets. The learning algorithm is applied to each training set and meta-learning is used to form a composite learner from the resulting classifiers.

A similar proposal is presented by Molinara et al. (2007). The proposed method builds a multiple classifier system where each constituting classifier is trained on a different subset of the majority class and on the whole minority class. The final classification system is obtained by combining all the single trained classifiers. This approach tries to avoid known drawbacks as overfitting of the minority class or incompleteness of the majority class.

As we have mentioned, it is difficult to determine the optimal amount of under- and/or over-sampling to apply and which of the techniques is more effective, i.e, the best way to tune the re-sampling paradigm is not an easy task. This problem was addressed by Estabrooks and Japkowicz (2001) and Estabrooks et al. (2004) and it was concluded that: a perfectly balanced data set is not necessarily optimal; and the best re-sampling rate varies. The conclusions of these works motivated the proposal of a mixture-of-experts framework (Estabrooks et al., 2004) as an effective solution to the tuning problem. This framework combines different expressions of the re-sampling approach on three levels: output level, expert level and classifier level. The output level combines the results of the over-sampling and under-sampling experts located at the expert level, which themselves each combine the results of 10 classifiers located at the classifier level and that resulted from learners trained on data sets sampled at different rates. The mixture-of-experts performs generally better than any re-sampling method that re-samples blindly to full balance. The proposed method was also found to perform better than both a single learner and a good-performing combination method such as Adaboost, on class imbalanced problems.

This idea of combining learners was also proposed in the work of Kotsiantis and Pintelas (2003). The authors use three agents (the first learns using Naive Bayes, the second using C4.5 and the third using 5NN) on a filtered version of training data and combine their predictions according to a voting scheme. A Facilitator agent is responsible for filtering the features of the data set and passing a copy of the instances into the three learning agents. Then, each learning agent re-samples data sets and returns prediction for each instance back to the Facilitator. The Facilitator makes the final prediction according to majority voting.

Del Castillo and Serrano (2004) present a complete and more sophisticated framework for addressing the problem of imbalanced data sets for the digital text categorization task. This framework incorporates feature selection and genetic algorithms in an architecture which is a combination of a variable number of learners. Learners may be added or removed depending on the specific text categorizationtask. This makes the system adaptable to any particular setting. A multi-strategy classifier system is used to construct multiple learners, each doing its own feature selection based on genetic algorithm. The predictions of each learner are combined using genetic algorithms.

4.5.2 Re-sampling and Algorithm Modifications

Re-sampling strategies were frequently integrated with algorithm modifications, specially with ensembles. We will briefly describe this widely explored area for classification tasks which involves the use of at least one pre-processing step and an adaptation of an algorithm.

SMOTE algorithm is combined with Complementary Neural Networks (CMTNN) in the work of Jeatrakul et al. (2010). CMTNN is a technique using a pair of complementary feedforward backpropagation neural networks called Truth Neural Network (TNN) and Falsity Neural Network (FNN). The TNN is trained to predict the degree of the truth memberships while the FNN is trained to predict the degree of false memberships. The strategy proposed by Jeatrakul et al. (2010) uses CMTNN to under-sample the training set and SMOTE to perform over-sample.

Random Forests are a well known ensemble type. Chen et al. (2004) proposes a method for dealing with highly-skewed class distributions based on the Random Forest algorithm. Balanced Random Forest (BRF) uses under-sampling of the majority class to create a training set with a more equal distribution between the two classes.

Some attention has also been given to SVMs, leading to proposals such as the one of Kang and Cho (2006) where an ensemble of under-sampled SVMs is presented. Multiple different training sets are built by sampling patterns from the majority class and combining them with the minority class patterns. Each training set is used for training an individual SVM classifier. The ensemble is produced by aggregating the outputs of all individual classifiers. Another similar approach is the EnSVM (Liu et al., 2006) which adopts a rebalance strategy combining SMOTE algorithm and under-sampling with an ensemble of SVMs. In a more recent work, Wang and Japkowicz (2010) proposes an ensemble of SVMs with asymmetric misclassification costs. The proposed system works by modifying the base classifier (SVM) using costs and uses boosting as the combination scheme.

A diverse set of approaches exist for embedding data pre-processing methods into boosting algorithms. In each iteration these algorithms change the weight distribution used to train the next learner towards the minority class. Examples within this type of approaches are: SMOTEBoost (Chawla et al., 2003), DataBoost-IM (Guo and Viktor, 2004b), JOUS-Boost (Mease et al., 2007), MSMOTEBoost (Hu et al., 2009), RamoBoost (Chen et al., 2010), RUSBoost (Seiffert et al., 2010) and EUSBoost (Galar et al., 2013). SMOTEBoost and MSMOTEBoost methods integrate respectively SMOTE and MSMOTE with Adaboost.M2 algorithm. To prevent boosting from overfitting, these algorithms do not update the weights associated with each example. Instead, they change the distributions by adding at each boosting iteration new synthetic examples of the minority class using the SMOTE and MSMOTE algorithm respectively. RUSBoost algorithm acts by removing instances from the majority class by random under-sampling the data set in each iteration. A new strategy was recently presented in Hulse et al. (2012) which modifies the RUSBoost algorithm improving its ability for also dealing with noise. This approach incorporates in RUSBoost the noise-handling capability of ORBoost algorithm Karmaker and Kwek (2006) to improve its performance with noisy data. DataBoost-IM uses the techniques described in Guo and Viktor (2004a) to generate new data examples integrating them with Adaboost.M1 algorithm. The major difference between this method and other boosting approaches with data generation is that it first identifies hard to learn examples and then carries out a rebalance process for both classes. The method called over/under-sampling with jittering (JOUS-Boost), uses random over-sampling and then introduces small perturbations into this data. Thus, at each boosting iteration, the algorithm uses synthetic data generated by the introduction of noise into the minority class examples obtained from random over-sampling. RamoBoost is a Ranked Minority Over-sampling technique based on the idea of adaptive synthetic data generation in a boosting learning system. The key idea is to adaptively rank minority class instances at each learning iteration according to a sampling probability distribution which is based on the underlying data distribution, and then adaptively shift the decision boundary towards the difficult instances. EUSBoost algorithm (Galar et al., 2013) is a recent contribution to this problem involving an evolutionary under-sampling guided boosting approach.

Also the integration of bagging and data pre-processing techniques can be considered. This is an usually simpler task than that of boosting. In fact, with a bagging learning system it is not required to compute new weights neither it is necessary to adapt any weight update formula. The most important task is to determine how each bootstrap replica is obtained. Several solutions exist for bagging learners embedding a diversity of sampling techniques. Regarding the integration of over-sampling techniques with bagging learning system it is straightforward to apply an over-sampling procedure in each bag before training the classifier. OverBagging (Wang and Yao, 2009) and SmoteBagging (Wang and Yao, 2009) are examples of this approach. Under-sampling methods has also been considered in this context existing a large diversity of approaches including under-sampling and bagging learning. Examples of this type of approaches are: QuasiBagging Chang et al. (2003), Asymmetric Bagging Tao et al. (2006), Roughly Balanced Bagging Hido et al. (2009), Partitioning Yan et al. (2003), UBagging Liang and Cohn (2013) and Bagging Ensemble Variation Li (2007). All these proposals maintain the same functional structure of incorporating an under-sampling technique for building each bag and using a bagging strategy. Some important differences among these approaches concern the construction of balanced or unbalanced bags for each iteration or the use of bags of varying size. The integration of bagging learning and a combination of both over-sampling and under-sampling strategies was also considered, being UnderOverBagging Wang and Yao (2009) a representative example. The Imbalanced IVotes (IIVotes) Błaszczyński et al. (2010) proposal combines the SPIDER data pre-processing method with IVotes.

Some more complex approaches combine pre-processing techniques with bagging and boosting, simultaneously, composing an ensemble of ensembles. EasyEnsemble and BalanceCascade algorithms (Liu, 2009) are examples of this approach type. Both algorithms use bagging as the main ensemble method and use Adaboost for training each bag. As for the pre-processing technique, both construct balanced bags by randomly under-sampling examples from the majority class. In EasyEnsemble algorithm all Adaboost iterations can be performed simultaneously once no operation is required after them. On the other hand, in BalanceCascade algorithm, after the Adaboost learning, the majority examples correctly classified with higher confidence are discarded from further iterations. For a more complete review on ensembles for the class imbalance problem see Galar et al. (2012).

4.5.3 Other Hybrid Strategies

A clustering method based on class purity maximization is proposed by Yoon and Kwek (2005). This method generates clusters of pure majority class examples and non-pure clusters based on the improvement of the clusters class purity. When the clusters are formed, all minority class examples are added to the non-pure clusters and a decision tree is built for each cluster. An unlabelled example is clustered according to the same algorithm. If it falls on a non-pure cluster, the decision tree committee votes the prediction, but if it fall on a pure majority class cluster the final prediction is produced (majority). If the committee votes for a majority class prediction, then that will be the final prediction, on the other hand if it is a minority class prediction, then the example will be submitted to a final classifier which is constructed using a neural network.

A strategy called SMOTE with different costs (SDC) was proposed by Akbani et al. (2004). It combines Smote with SVMs integrated with costs. The SVM is biased in a way that pushes the boundary away from the positive instances. To do that different error costs are used for the positive and negative classes. Using different error costs for different classes to push the boundary away from the positive instances. SMOTE is used to make the positive instances more densely distributed in order to make the boundary more well defined.

In Tahir et al. (2012) a novel inverse random under-sampling (IRUS) method is presented. The main idea is to repeatedly severely under-sample the majority class for creating a large number of distinct training sets.For each training set a decision boundary is found which separates the minority class from the majority class. By combining the multiple designs through fusion, a composite boundary between the majority class and the minority class is constructed. In Zhang et al. (2013) IRUS algorithm is combined with Random Tree. IRUS algorithm is used for generating multiple distinct training sets.Then, with each training set, a random tree is trained to separate the minority class from the majority class. By combining these random trees through fusion, a composite classifier is constructed.

Recently, Sumadhi and Hemalatha (2013) proposed a new technique called IFSMOTE which involves the combination of FSMOTE algorithm (presented in Section 4.2.1.6) for data generation and Adaboost algorithm. IFSMOTE new synthesised examples agree to the concept of fractal interpolation theory and the gentle Adaboost algorithm is used to improve the performance.

	Strategy type	Section	Main References
Re-sampling	Random Under/Over-sampling	4.2.1.1	Chawla et al. (2002); Drummond et al. (2003); Estabrooks et al. (2004)
	Distance Based	4.2.1.2	Chyi (2003); Mani and Zhang (2003)
	Data Cleaning Based	4.2.1.3	Kubat and Matwin (1997); Laurikkala (2001); Batista et al. (2004); Naganjaneyulu and Kuppa (2013)
	Cluster Based	4.2.1.4	Jo and Japkowicz (2004), Yen and Lee (2006, 2009), Cohen et al. (2006)
	Synthesising New Data	4.2.1.5	Lee (1999, 2000); Chawla et al. (2002); Liu et al. (2007); Menardi and Torelli (2010); Martínez-García et al. (2012)
	Adaptive Synthetic Sampling	4.2.1.6	Batista et al. (2004); Han et al. (2005); He et al. (2008); Bunkhumpornpat et al. (2009); Hu et al. (2009); Zhang et al. (2011); Maciejewski and Stefanowski (2011); Barua et al. (2012); Ramentol et al. (2012b,a); Verbiest et al. (2012); Bunkhumpornpat et al. (2012); Nakamura et al. (2013)
	Evolutionary Sampling	4.2.1.7	García et al. (2006a); Doucette and Heywood (2008); García and Herrera (2009); Drown et al. (2009); Maheshwari et al. (2011); Yong (2012); Derrac et al. (2012)
	Re-sampling combinations	4.2.1.8	Stefanowski and Wilk (2008); Napierała et al. (2010); Bunkhumpornpat et al. (2011); Songwattanasiri and Sinapiromsaran (2010); Vasu and Ravi (2011); Yang and Gao (2012); Li et al. (2008)
Active Learning		4.2.2	Ertekin et al. (2007b,a); Zhu and Hovy (2007); Ertekin (2013); Mi (2013)
Weighting the Data Space		4.2.3	Zadrozny et al. (2003)

Table 4.1: Pre-processing strategy types, corresponding sections and main bibliographic references

Strategy type	Section	Main References
Recognition-based	4.3.1	Chawla et al. (2004); Schölkopf et al. (2001); Manevitz and Yousef (2002); Raskutti and Kowalczyk (2004); Zhuang and Dai (2006); Lee and Cho (2006); Japkowicz et al. (1995, 2000); Japkowicz (2001b); Manevitz and Yousef (2007); Bellinger et al. (2012)
Cost-sensitive algorithms	4.3.2	Maloof (2003); Ling et al. (2004); Elkan (2001); Drummond and Holte (2000); Gama (2003); Veropoulos et al. (1999); Akbani et al. (2004); Tang et al. (2009); Yuanhong et al. (2009); Hwang et al. (2011); Weiguo et al. (2012); Zhou and Liu (2006); Alejo et al. (2007); Oh (2011); Castro and de Pádua Braga (2013); Cao et al. (2013); Fan et al. (1999); Ting (2000); Joshi et al. (2001); Sun et al. (2007); Song et al. (2009); Chen et al. (2004)
New algorithms	4.3.3	Wu and Chang (2003); Imam et al. (2006); Tang and Zhang (2006); Batuwita and Palade (2010b); Li et al. (2009); Barandela et al. (2003); Huang et al. (2004); Liu et al. (2010) Cieslak and Chawla (2008); Cieslak et al. (2012); Phua et al. (2004); Rodriguez et al. (2012); Wu and Chang (2005); Tan et al. (2003); Xiao et al. (2012); Ribeiro (2011)

Table 4.2: Strategies of algorithms modifications, corresponding sections and main bibliographic references

4.6 Summary

In this section we provide a global and systematic overview of the strategy types previously discussed. In Tables 4.1, 4.2, 4.3 and 4.4 we have a summary of the categorised strategies, the corresponding section and the main bibliographic references.

Strategy type	Section	Main References
Threshold method	4.4.1	Maloof (2003); Weiss (2004)
Cost-sensitive post-processing	4.4.1	Karagiannopoulos et al. (2007); Domingos (1999); Zadrozny and Elkan (2002); Sinha and May (2004); Bansal et al. (2008); Zhao et al. (2011); Hernández-Orallo (2012)

Table 4.3: Post-processing strategy types, corresponding sections and main bibliographic references

Strategy type	Section	Main References
Combinations of algorithms predictions	4.5.1	Chan and Stolfo (1998); Molinara et al. (2007); Estabrooks et al. (2004); Kotsiantis and Pintelas (2003); Del Castillo and Serrano (2004)
Re-sampling and algorithm modifications	4.5.2	Jeatrakul et al. (2010); Chen et al. (2004); Kang and Cho (2006); Liu et al. (2006); Chawla et al. (2003); Guo and Viktor (2004b); Mease et al. (2007); Hu et al. (2009); Chen et al. (2010); Seiffert et al. (2010); Galar et al. (2013); Wang and Yao (2009); Chang et al. (2003); Tao et al. (2006); Hido et al. (2009); Yan et al. (2003); Liang and Cohn (2013); Li (2007); Błaszczyński et al. (2010); Liu (2009); Wang and Japkowicz (2010)
Other	4.5.3	Yoon and Kwek (2005); Akbani et al. (2004); Tahir et al. (2012); Zhang et al. (2013); Sumadhi and Hemalatha (2013)

Table 4.4: Hybrid strategies, corresponding sections and main bibliographic references

Chapter 5

Re-sampling for Regression

5.1 Introduction

Among the different types of existing approaches to handle imbalanced distributions the re-sampling methods are the most simple and versatile. These methods change the data distribution and, therefore, allow the use of any standard learning algorithm. Still, only for classification tasks these approaches have been extensively studied, and no work exists within the regression setting.

We describe three different re-sampling methods for regression tasks under imbalanced domains. We start by the simplest of all, which is random under-sampling. Then, we introduce an adaptation of the well-known and successful Smote algorithm to regression tasks, which we named SmoteR . Finally, we propose the adaptive sampling algorithm, which is a method less dependent on the user and thus, a more flexible approach. All these methods address the problem of predicting rare extreme values of a continuous variable and depend on a user-defined relevance function ($\phi()$) which expresses the user preference bias.

5.2 Random Under-sampling

The first strategy we propose is random under-sampling.The basic idea of under-sampling (e.g. Kubat and Matwin (1997)) is to decrease the number of observations with the most common target variable values with the goal of better balancing the ratio between these observations and the ones with the interesting target values, which are less frequent. Within classification this consists on obtaining a random sample from the training cases with the frequent (and less interesting) class values. This sample is then joined with the observations with the rare target class value to form the final training set that is used by the selected learning algorithm. This means that the training sample resulting from this approach will be smaller than the original (imbalanced) data set.

In regression we have a continuous target variable. As mentioned in Section 2.1, the notion of relevance can be used to specify the values of a continuous target variable that are more important for the user. We can also use the relevance function values to determine which are the observations with the common and uninteresting values that should be under-sampled. Namely, we propose to randomly under-sampling observations whose target value has a relevance less than a user-defined threshold t_R. Under-sampling will be carried out on the set $\mathcal{D}_N = \{\langle \mathbf{x}_i, y_i \rangle \in \mathcal{D} : \phi(y_i) \leq t_R\}$ which contains the more frequent and uninteresting observations for the user. The selected observations are then joined with the set $\mathcal{D}_R = \mathcal{D} \setminus \mathcal{D}_N$.

Regarding the amount of under-sampling that is to be carried out the strategy is the following. For each of the relevant observations in $\mathcal{D}_R$ we will randomly select n_u cases from the "normal" observations in $\mathcal{D}_N$. The value of n_u is another user-defined parameter that will establish the desired ratio between "normal" and relevant observations. Too large values of n_u will result in a new training data set that is still too unbalanced, but too small values may result in a training set that is too small, particularly if there are too few relevant observations.

As an example of the possible consequences of this strategy to a domain, suppose a given data set has 100 observations and $|\mathcal{D}_R| = 20$ for a certain threshold t_R considered by the user. On this setting, if the parameter n_u is 2, this means that, for each example in $\mathcal{D}_R$, two examples will be randomly selected from the set $\mathcal{D}_N$, producing a new data set with a total of 60 examples (20 rare cases and 40 normal cases). We will have a new data set with twice as much examples from the $\mathcal{D}_N$ set than from $\mathcal{D}_R$. On the other hand, if n_u is set to 0.5 the normal cases will be 50% of the rare cases, i.e., for the same data set with 100 observations, and 20 rare cases, only 10 examples will be selected from the normal ones, producing a new data set with only 30 examples and an unbalance favouring the $\mathcal{D}_R$ cases.

5.3 The SmoteR Algorithm

Smote (Chawla et al., 2002) is a sampling method to address classification problems with imbalanced class distribution. As we have mentioned in Section 4.2.1.5 the key feature of this method is the combination of under-sampling of the majority class with an innovative over-sampling strategy which involves the generation of synthetic examples for the minority class. We propose a variant of Smote for addressing regression tasks where the key goal is to accurately predict rare extreme values, which we will name SmoteR .

The original Smote algorithm uses an over-sampling strategy that consists on generating "synthetic" cases with a rare target value. Chawla et al. (2002) propose an interpolation strategy to create these artificial examples. For each case from the set of observations with rare values ($\mathcal{D}_R$), the strategy is to randomly select one of its k-nearest neighbours from this same set. With these two observations a new example is created whose attribute values are an interpolation of the values of the two original cases. Regards the target variable, as Smote is applied to classification problems with a single class of interest, all cases in $\mathcal{D}_R$ belong to this class and the same will happen to the synthetic cases. For handling data sets with both numeric and nominal features the Smote -NC algorithm is proposed (Chawla et al., 2002). In this case, for determining the k-nearest neighbours of an example a penalisation is also introduced for the nominal features. The nominal feature values of the new synthetic example are decided according to the value occurring in the majority of the k-nearest neighbours.

There are three key components of the Smote algorithm that we need to address in order to adapt it for our target regression tasks: i) how to define which are the relevant observations and the "normal" cases; ii) how to create the attribute values of the new synthetic examples (i.e. over-sampling); and iii) how to decide the target variable value of these new synthetic examples. Regarding the first issue, the original algorithm is based on the information provided by the user concerning which class value is the target/rare class (usually known as the minority or positive class). In our problems we face a potentially infinite number of values of the target variable. Our proposal is based on the existence of a relevance function ($\phi(y)$) and on a user-specified threshold on the relevance values (t_R), that leads to the definition of the set $\mathcal{D}_R$. Our algorithm will over-sample the observations in $\mathcal{D}_R$ and randomly under-sample the cases in $\mathcal{D}_N$, thus leading to a new training set with a redefined distribution of the target values.

Regarding the second key component, the generation of the attributes of the new cases, we use the same interpolation approach as in the original algorithm for numeric features. For handling nominal attributes we have introduced some small modifications. We simplified the way Smote -NC strategy handles nominal attributes. For creating a nominal feature value of a new case we randomly select one of the feature values of the two examples used for generating the new one.

Finally, the third key issue is to decide the target variable value of the generated observations. In the original algorithm this is a trivial question, because as all rare cases have the same class (the target minority class), the same will happen to the examples generated from this set. In our case the answer is not so trivial. The cases that are to be over-sampled do not have the same target variable value, although they do have a high relevance score ($\phi(y)$). This means that when a pair of examples is used to generate a new synthetic case, they might not have the same target variable value. Our proposal is to use a weighted average of the target variable values of the two seed examples. The weights are calculated as an inverse function of the distance of the generated case to each of the two seed examples.

Algorithm 5.1 The main SmoteR algorithm.

```
function SmoteR($\mathcal{D}, t_R, o, u, k$)
    // $\mathcal{D}$ - A data set
    // $t_R$ - The threshold for relevance of the target variable values
    // $\%o$,$\%u$ - Percentages of over- and under-sampling
    // $k$ - The number of neighbours used in case generation

    $rareL \leftarrow \{\langle \mathbf{x}, y \rangle \in \mathcal{D} : \phi(y) > t_R \wedge y < \tilde{y}\}$  // $\tilde{y}$ is the median of the target $Y$
    $newCasesL \leftarrow$ genSynthCases($rareL, \%o, k$)  // generate synthetic cases for rareL
    $rareH \leftarrow \{\langle \mathbf{x}, y \rangle \in \mathcal{D} : \phi(y) > t_R \wedge y > \tilde{y}\}$
    $newCasesH \leftarrow$ genSynthCases($rareH, \%o, k$)  // generate synthetic cases for rareH
    $newCases \leftarrow newCasesL \bigcup newCasesH$
    $nrNorm \leftarrow \%u$ of $|newCases \bigcup rareL \bigcup rareH|$
    $normCases \leftarrow$sample of $nrNorm$ cases $\in \mathcal{D} \setminus \{rareL \bigcup rareH\}$  // under-sampling
    return $newCases \bigcup normCases$
end function
```

Algorithm 5.1 describes our proposed SmoteR sampling method. The algorithm uses a user-defined threshold (t_R) of relevance to define the sets $\mathcal{D}_R$ and $\mathcal{D}_N$ of relevant and normal respectively. Notice that, in our target applications, we may have two rather different sets of rare cases: the extreme high and low values. This is another

Algorithm 5.2 Generating synthetic cases.

```
function genSynthCases($\mathcal{D}, o, k$)

    $newCases \leftarrow \{\}$
    $ng \leftarrow \%o/100$  // nr. of new cases to generate for each existing case
    for all $case \in \mathcal{D}$ do
        $nns \leftarrow$ kNN($k, case, \mathcal{D}_r \setminus \{case\}$)  // k-Nearest Neighbours of case
        for $i \leftarrow 1$ to $ng$ do
            $x \leftarrow$ randomly choose one of the $nns$
            for all $a \in$ attributes do  // Generate attribute values
                if isNumeric($a$) then
                    $diff \leftarrow case[a] - x[a]$
                    $new[a] \leftarrow case[a] +$ random$(0, 1) \times diff$
                else
                    $new[a] \leftarrow$ randomly select among $case[a]$ and $x[a]$
                end if
            end for
            $d_1 \leftarrow$ dist($new, case$)  // Decide the target value
            $d_2 \leftarrow$ dist($new, x$)
            $new[Target] \leftarrow \frac{d_2 \times case[Target] + d_1 \times x[Target]}{d_1 + d_2}$
            $newCases \leftarrow newCases \bigcup \{new\}$
        end for
    end for
    return $newCases$
end function
```

difference to the original algorithm. The consequence of this is that the generation of the synthetic examples is also done separately for these two sets. The reason is that although both sets include rare and interesting cases, they are of different type and thus with very different target variable values (extremely high and low values). The other parameters of the algorithm are the percentages of over- and under-sampling, and the number of neighbours to use in the cases generation. The key aspect of this algorithm is the generation of the synthetic cases. This process is described in detail on Algorithm 5.2. The main differences to the original SMOTE algorithm are: the way nominal variables are handled; and the way the target value for the new cases is generated. Regards the former issue we simply perform a random selection between the values of the two seed cases. A possible alternative could be to use some biased sampling that considers the frequency of occurrence of each of the values within the rare cases. Regarding the target value we have used a weighted average between the values of the two seed cases. The weights are decided based on the distance between the new case and these two seed cases. The larger the distance, the smaller the weight.

This strategy changes the distribution of rare and normal cases in a, sometimes, drastic way. For instance, consider an hypothetical domain with 100 observations and $|D_R| = 20$ for a given relevance threshold t_R. In this setting, if the SMOTER algorithm is applied with 200% for over-sampling and 50% for under-sampling, the new data set will have a total of 90 examples now distributed as follows: 60 rare cases (20 original rare + 40 synthetic rare cases) and 30 normal cases (50% of the 60 rare examples). A more extreme example of the effects of this strategy can be seen when we consider the same setting now with 700% for over-sampling and 200% for under-sampling. This will result in a new data set with a total of 480 examples: 160 ($20 + 7 \times 20 = 160$) rare cases and 320 ($160 \times 2 = 320$) normal cases.

5.4 The Adaptive Sampling Algorithm

The adaptive samplingmethod is a sampling strategy for addressing the problem of predicting rare extreme values of a continuous variable. As the previous method, this approach is also based on a user-defined relevance function ($\phi(y)$) which is used to determine: where to perform over-/under-sampling, and the amount of cases to be generated/eliminated. The main goal of adaptive sampling is to ensure that the training sample provided to the learning algorithm will reflect the preference biases of the user expressed with the relevance function. Adaptive sampling has the advantage of minimising the user intervention while maintaining the capability of using any standard regression learner. To apply this method the user does not need to select neither the relevance threshold t_R nor the over- and under-sampling percentages. The general idea of the proposed method is to use the relevance function to discretize the target variable values into bins. For each constructed bin, a target frequency is calculated from the relevance function. This target frequency aims to obtain a distribution of examples towardsthe user preferences, increasing the number of examples in the more important bins and decreasing that number in the less interesting bins. A strategy for over-/under-sampling is applied as needed inside each bin. This way it is possible to apply different strategies over the target variable range, adjusting the distribution of the training set to the user preferences.

Algorithm 5.3 The main Adaptive Sampling algorithm.

```
function AdaptiveSampling($\mathcal{D}, N, d$)
    // $\mathcal{D}$ - A data set $\{\mathrm{x}, y_i\}_{i=1}^{n}$
    // $N$ - Number of intervals into which the relevance values will be discretized
    // $d$ - Disturbance applied when generating examples with Gaussian noise

    $Bs \leftarrow$ binsConstructor($\mathcal{D}[Target], \phi, N$)   // examples indexes and the mean relevance in each bin
    $k \leftarrow |Bs|$   // number of bins generated
    $newCases \leftarrow \{\}$
    $tot\phi \leftarrow \sum_{b \in Bs} b.\overline{\phi}$
    for $b \in Bs$ do
        $p(b) \leftarrow |b.exs|$
        $w(b) \leftarrow \frac{b.\overline{\phi}}{tot\phi}$
        $\hat{p}(b) \leftarrow w(b) \times |\mathcal{D}|$
        if $\hat{p}(b) > p(b)$ then   // Apply the oversampling strategy
            $synth \leftarrow$ genPerturb($b.exs, \hat{p}(b), \mathcal{D}, d$)
            $newCases \leftarrow newCases \bigcup \mathcal{D}[b.exs] \bigcup synth$
        else
            if $\hat{p}(b) < p(b)$ then   // Apply random under-sampling
                $newCases \leftarrow newCases \bigcup$ sample( $\mathcal{D}[b.exs]$, $\hat{p}(b)$ )
            else   // Just add the examples in the bin
                $newCases \leftarrow newCases \bigcup \mathcal{D}[b.exs]$
            end if
        end if
    end for
    return $newCases$
end function
```

Algorithm 5.3 describes our proposed adaptive sampling method. This strategy consists of three steps: i) construct bins over the target variable domain; ii) calculate

Algorithm 5.4 Algorithm for constructing the data set bins.

```
function BinsConstructor($Tgt, \phi, N$)
    // $Tgt$ - The data set target values
    // $\phi$ - The relevance function
    // $N$ - number of intervals into which the relevance values will be discretized

    $OrdTgt \leftarrow \mathrm{ORDER}(Tgt)$
    $RelTgt \leftarrow \{\phi(x) : x \in OrdTgt\}$
    $\delta \leftarrow \frac{1}{N}$   // Relevance variation in each interval
    for $i \leftarrow 1$ to $N$ do
        $MeanRelev[i] \leftarrow \frac{\delta}{2} + (i-1) \times \delta$   // Mean relevance in each interval
    end for
    $cutsTgt \leftarrow \mathrm{cut}(RelTgt, N)$   // Match each relevance value with the corresponding interval
    $Bs \leftarrow \{\}$
    $b.exs \leftarrow \{\}$
    $currCT \leftarrow cutsTgt[1]$
    for $i \in OrdTgt$ do
        if $cutsTgt[i] = currCT$ then
            $b.exs \leftarrow b.exs \bigcup \{i\}$
        else
            $b.\overline{\phi} \leftarrow MeanRelev[currCT]$
            $Bs \leftarrow Bs \bigcup \{ (b.exs, b.\overline{\phi}) \}$
            $b.exs \leftarrow \{i\}$
            $currCT \leftarrow cutsTgt[i]$
        end if
    end for
    $Bs \leftarrow Bs \bigcup \{ (b.exs, MeanRelev[currCT]) \}$
    return $Bs$
end function
```

Algorithm 5.5 Algorithm for generating synthetic examples with Gaussian Noise.

```
function genPerturb($ind, obj, \mathcal{D}, d$)
    // $ind$ - The indexes of the examples in a given bin
    // $obj$ - The number of examples to obtain in the bin
    // $\mathcal{D}$ - The data set
    // $d$ - Disturbance applied when generating examples with Gaussian noise

    $freq \leftarrow$ frequency of nominal attributes in $\mathcal{D}$
    $sd \leftarrow$ standard deviation of numeric attributes in $\mathcal{D}$
    $nr.att \leftarrow$ MAX($\sqrt{attrs},\ attrs \times 10\%$)  // Number of attributes to perturb
    $ng \leftarrow$ nr of synthetic examples to generate for each existing example
    $new \leftarrow \{\}$
    for all $case \in \mathcal{D}[ind]$ do
        for $i \leftarrow 1$ to $ng$ do
            $sel.att \leftarrow$ SAMPLE($attrs, nr.att$)  // Randomly select $nr.att$ attributes to perturb
            for all $a \in attributes$ do  // Generate attribute values for the new case
                if isNumeric($a$) and $a \in sel.att$ then
                    $new[a] \leftarrow case[a] +$ RNORM($0, d \times sd[a]$)
                else
                    if isNominal($a$) and $a \in sel.att$ then
                        $new[a] \leftarrow$ SAMPLE($nom.att, 1, probs = freq[a]$)
                    else
                        $new[a] \leftarrow case[a]$
                    end if
                end if
            end for
            $new[Tgt] \leftarrow case[a] +$ RNORM($0, d \times sd[Tgt]$)  // Generate target value
        end for
    end for
    return $new$
end function
```

the target frequency of each bin; iii) adapt the data set frequencies by applying an over-sampling or under-sampling strategy as required by the target frequencies.

On a first phase, the bins are adaptively built based on the user-defined relevance function ϕ, and a parameter N that controls the number of intervals into which the relevance values will be discretized. The process of generating the data set bins is described on Algorithm 5.4. This algorithm starts by sorting the target variable values (y_i) and assigning to each $\phi(y_i)$ a number ranging from 1 to N which corresponds to the interval the example is in. This is done using the function CUT(). Having each interval assigned, the algorithm goes through all sorted y_i and stores the examples indexes of each constructed bin and the corresponding bin mean relevance. We highlight that the number N of intervals considered may not match the number of bins generated which we represent by k. In effect, if the data set has two extreme types (low and high) it is possible to generate at most a total of $2 \times N - 1$ different bins. This relation between the number of intervals and the number of bins generated is exemplified in Figure 5.1 for the *fuelCons* data set, where the assigned bins are represented by coloured lines at the bottom.

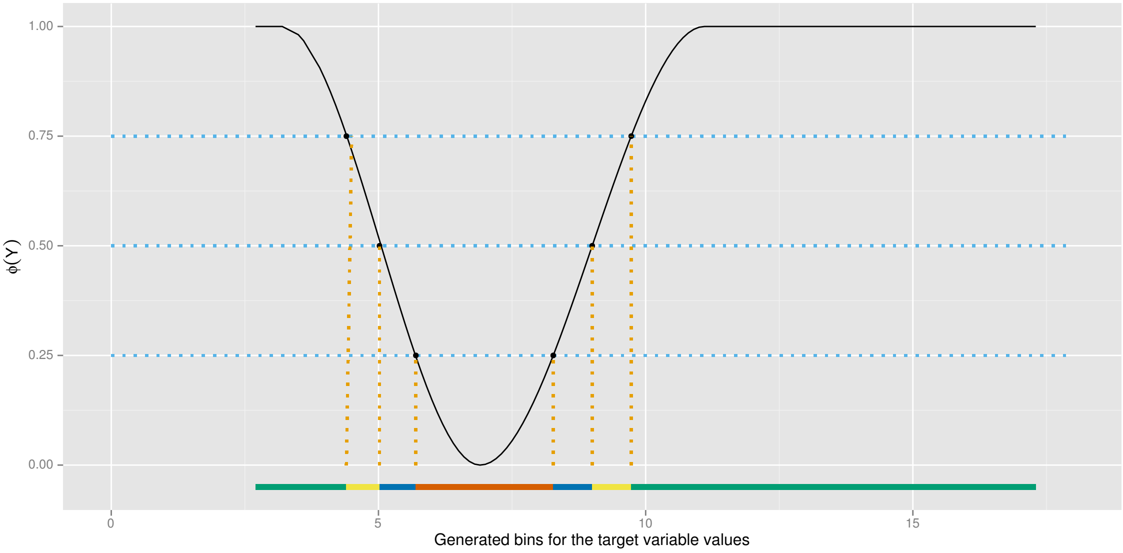


Figure 5.1: Split of data set *fuelCons* with 4 intervals according to $\phi()$ value and the generated bins after applying the Algorithm 5.4.

Regarding the second step, for each bin the observed frequency is calculated, and a new target frequency is estimated, approximately maintaining the total number of examples of the data set $(|\mathcal{D}|)$. To do so, each bin b_i is assigned a relative importance

$w(b_i)$ defined as:

$$w(b_i) = \frac{\phi(b_i)}{\sum_{j=1}^{k} \phi(b_j)} \tag{5.1}$$

Using this relative importance the target frequency $\hat{p}(b_i)$ of each bin is estimate as,

$$\hat{p}(b_i) = w(b_i) \times |\mathcal{D}| \tag{5.2}$$

An example of this procedure can be seen in Figure 5.2

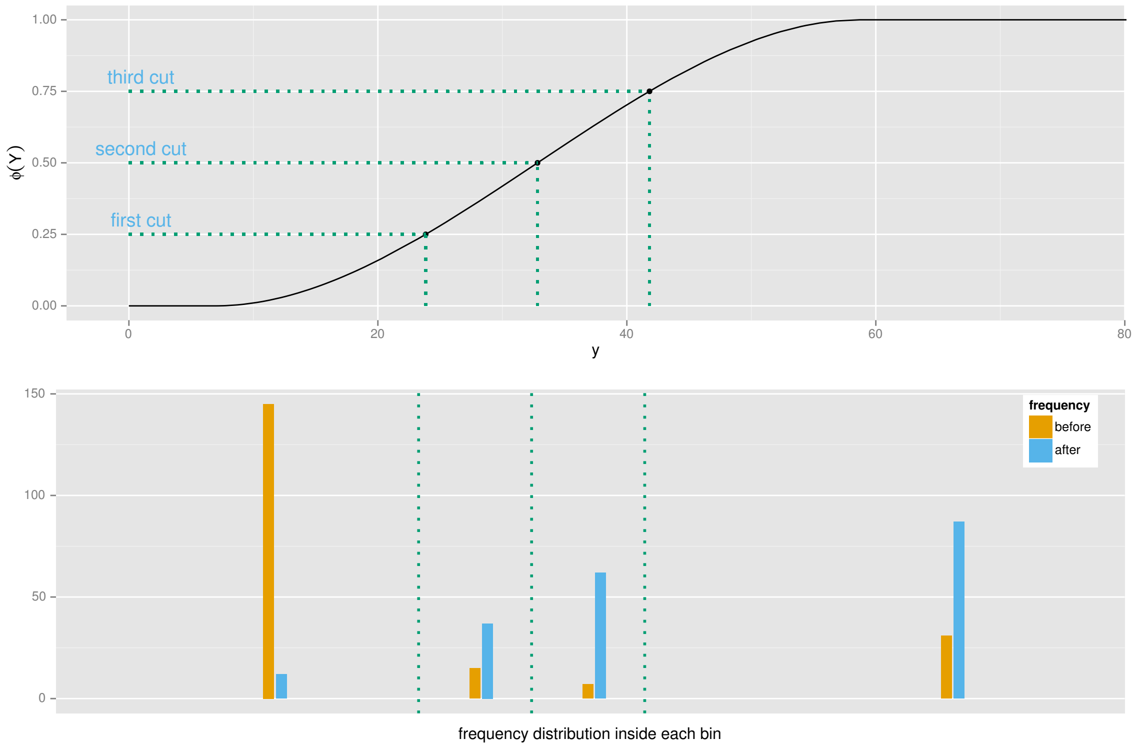


Figure 5.2: Split of data set *a1* with 4 intervals according to $\phi()$ value, and the examples frequency of each built bin before and after applying the adaptive sampling algorithm.

Finally, for the third step, having the observed frequency and the target frequency of each bin, a re-sampling strategy is used as appropriate: if the target frequency is larger than the observed frequency over-sampling is applied, otherwise an under-sampling strategy is used. We decided to use random under-sampling due to its simplicity. Regarding the over-sampling strategy a new method for generating synthetic examples based on the introduction of Gaussian noise was chosen. This algorithm creates synthetic examples using the original examples provided in the bin and a parameter d which controls the radius of the neighbourhood where the synthetic cases will be generated.

Each example is generated by the introduction of a small perturbation on a small number of randomly selected attributes. The algorithm is able to deal with both types of attributes: for the nominal attributes a value is chosen accordingly to the observed frequency on the data set; the numeric attributes are perturbed by the introduction of Gaussian noise based on the attribute standard deviation and weight d. The target variable value is also perturbed in a similar way as numeric attributes, although in this case the neighbourhood of the perturbation allowed is narrower, and the target variable values are guaranteed to be within the considered bin range. This method is described in detail on Algorithm 5.5.

Although adaptive sampling is meant to address the problem of predicting rare extreme values, it can actually be used within a broader scenario. The relevance function defined by the user is the key aspect for determining the more important values which may not be extremes and also do not need to be less frequent. This approach can be applied to situations of cost based applications where some values, though important, are already frequent on the given training data. Obviously, this would mean that the impact of this method would not be so significant. Still, we could make some adjustments to the examples frequency by applying this method to better adjust the training data to the user preference biases.

5.5 Experimental Analysis

5.5.1 Experimental Setup

The goal of our experiments is to test the effectiveness of our proposed sampling approaches at predicting rare extreme values of a continuous target variable. For this purpose, we have selected 18 regression data sets. Most of these data sets come from Torgo's repository of regression problems[1], where further details can be obtained on these tasks. The 7 algae tasks ($a1 \cdots a7$) are from an international data analysis competition[2] and the data as well as a description can be obtained

[1] http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html

[2] http://www.erudit.de/erudit/

Data Set	N	p	Ext	threshold=0.75		threshold=0.9		threshold=0.95	
				$nRare$	$\%Rare$	$nRare$	$\%Rare$	$nRare$	$\%Rare$
a1	198	12	H	31	0.157	22	0.111	20	0.101
a2	198	12	H	24	0.121	15	0.076	15	0.076
a3	198	12	H	34	0.172	30	0.152	26	0.131
a4	198	12	H	34	0.172	27	0.136	21	0.106
a5	198	12	H	22	0.111	18	0.091	16	0.081
a6	198	12	H	33	0.167	28	0.141	28	0.141
a7	198	12	H	27	0.136	27	0.136	26	0.131
Abalone	4177	9	H/L	679	0.163	564	0.135	438	0.105
Accel	1732	15	H	102	0.059	61	0.035	52	0.03
dAiler	7129	6	H/L	450	0.063	267	0.037	230	0.032
availPwr	1802	16	H	169	0.094	141	0.078	131	0.073
bank8FM	4499	9	H	339	0.075	198	0.044	144	0.032
cpuSm	8192	13	L	755	0.092	616	0.075	541	0.066
dElev	9517	7	H/L	1109	0.117	478	0.05	310	0.033
fuelCons	1764	38	H/L	200	0.113	105	0.06	89	0.05
heat	7400	13	H	729	0.099	525	0.071	453	0.061
boston	506	14	H	69	0.136	53	0.105	51	0.101
maxTorque	1802	33	H	158	0.088	92	0.051	87	0.048

Table 5.1: Used data sets and characteristics (N: nr. of cases; p: nr. of predictors; Ext: extreme type H (high)/ L (Low); $nRare$: nr. cases with $\phi(Y) > threshold$; $\%Rare$: $nRare/N$).

in Torgo (2010). Finally, the *availPwr*, *fuelCons*, *maxTorque* and *Heat* data sets are from the automotive industry, but no further information can be disclosed given their commercial nature. Table 5.1 shows the main characteristics of these data sets.

For each data set it is necessary to define which values are the extreme and more important ones. This would require the intervention of an expert on each of the domains for defining the corresponding relevance function. To solve this problem we have obtained a relevance function using the automatic method proposed by Ribeiro (2011). This method assigns higher relevance for values above (below) the adjacent values of the target variable distribution. These are calculated as a function of the quartiles and the inter-quartile range that are well-known thresholds for considering a value as an outlier. The result of this method are relevance functions that assign higher relevance to high and low rare extreme values, which constitute our goal.

Based on these relevance functions, we have decided to test different thresholds on the values of $\phi(Y)$ as the condition for a value to be taken as a rare extreme. We have considered the three following values for threshold: 0.75, 0.9 and 0.95. This will allows

Learner	Parameter Variants	R package
MARS	$nk = \{10, 17\}, degree = \{1, 2\}, thresh = \{0.01, 0.001\}$	**earth** Milborrow (2012)
SVM	$cost = \{10, 150, 300\}, gamma = \{0.01, 0.001\}$	**e1071** Dimitriadou et al. (2011)
Random Forest	$mtry = \{5, 7\}, ntree = \{500, 750, 1500\}$	**randomForest** Liaw and Wiener (2002)

Table 5.2: Regression algorithms and parameter variants, and the respective R packages.

us to evaluate the performance of our proposed methods for domains with different percentages of rare cases. As it can be seen from the information in Table 5.1, the higher the threshold considered, the fewer rare cases exist in each domain. For the 18 data sets used in our experiments, this results in an average percentage of the available cases having a rare extreme value of: 11.9% for threshold 0.75; 8.8% for threshold 0.9 and 7.8% for threshold 0.95. Considering several relevance thresholds will enable us to compare the impact of the proposed sampling strategies in domains with different rare cases percentages.

In order to avoid any algorithm-dependent bias distorting our results, we have carried out our comparisons using a diverse set of standard regression algorithms. Moreover, for each algorithm we have considered several parameter variants. Table 5.2 summarises the learning algorithms that were used and also the respective parameter variants. The combination of all the parameter values reported in Table 5.2 results in 8 variants of the Multivariate Adaptive Regression Spline (MARS) regression algorithm (Friedman, 1991), 6 variants of the Support Vector Machine (SVM) algorithm (Cortes and Vapnik, 1995) and 6 variants of the Random Forest algorithm (Breiman, 2001). To ensure easy replication of our work we have used the implementations of these algorithms available in the free open source R environment (R Core Team, 2013), which is also the infrastructure used to implement our proposed re-sampling methods.

Each of the 20 learning approaches (8 MARS variants + 6 SVM variants + 6 Random Forest variants), were applied to each of the 18 regression problems using 29 different sampling approaches. Sampling comprises the following approaches: i) carrying out no sampling at all (i.e. use the data set with the original imbalance); ii) 12 variants of SmoteR method; iii) 4 variants of under-sampling; and iv) 12 variants of adaptive sampling method. The 12 SmoteR variants used 5 nearest neighbours for case generation and all combinations of $\{25, 50, 100, 200\}\%$ and $\{200, 500, 700\}\%$ for percentages of under- and over-sampling, respectively. The 4 under-sampling variants used $\{25, 50, 100, 200\}\%$ for percentage of under-sampling. The 12 adaptive sampling variants used all combinations of $\{2, 4, 6, 8\}$ and $\{0.02, 0.05, 0.1\}$ for the number of

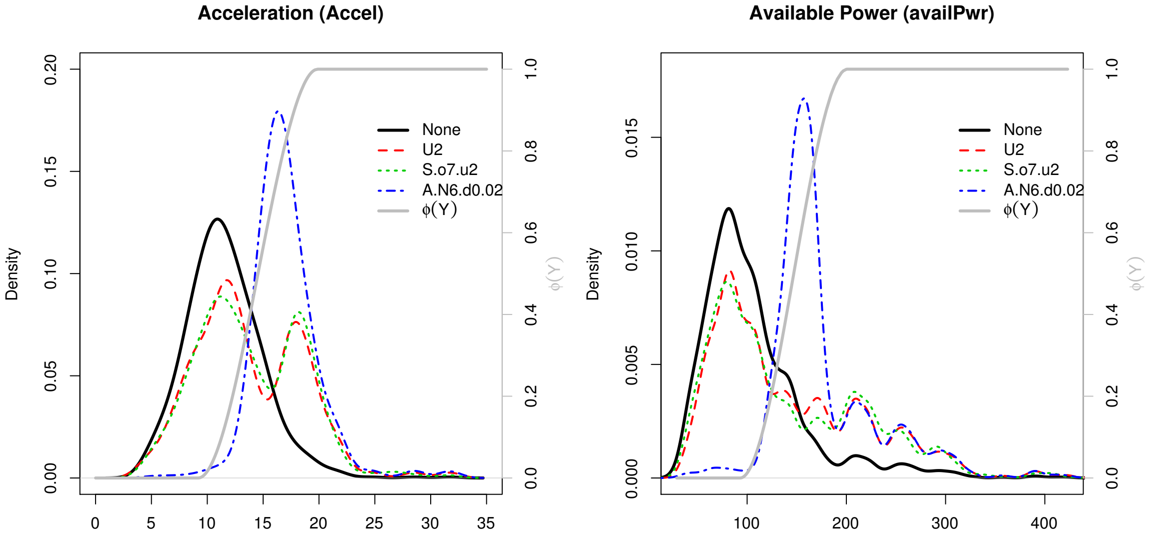


Figure 5.3: Distribution of the target variable before and after re-sampling for data sets *Accel* and *availPwr* with relevance threshold set to 0.75.

splits performed in the relevance interval and the allowed radius of the neighbourhood where the synthetic examples are generated respectively.

To have a better idea on the impact of these re-sampling strategies on the training set that is finally given to the regression tools, Figure 5.3 shows the distribution of the target variable[3] on the original data and on the data sets resulting from applying three of the most successful variants of our re-sampling strategies, for two specific data sets. The graphs in this figure clearly illustrate the change in the target variable distribution that is carried out by these re-sampling strategies with the goal of biasing this distribution towards the areas where the relevance function has higher values. Moreover, as previously mentioned, with the exception of the adaptive sampling, the methods also change the total number of cases used for training, which will obviously have an impact on the computation time taken to obtain the models. More specifically, for the data sets in Figure 5.3 and the relevance threshold of 0.75, the original *Acceleration* (*Accel*) data set contains 1732 observations and the *S.o7.u2* configuration of SmoteR leads to a training set of 2448, the *U2* under-sampling variant uses only 306 cases and finally the *A.N6.d0.02* variant of Adaptive Sampling results in a data set with 1732 examples. With respect to the *Available Power* (*availPwr*) data set the original size is 1802 and the same three re-sampling variants use 4056, 507 and 1801,respectively.

[3] Approximated through a kernel density estimator.

Our goal is to compare the 28 sampling strategies (12 SmoteR + 4 under-sampling + 12 adaptive sampling) against the default of using the given data, using 20 learning approaches and 18 data sets for each relevance threshold considered (0.75, 0.9 and 0.95). . All alternatives we have described were evaluated according to F_1, the F-measure with $\beta = 1$ (cf. Equation 3.9), which means that the same importance was given to both precision and recall scores that were calculated using the set-up described in Section 3.3. The values of the F-measure were estimated by means of 3 repetitions of a 10-fold cross validation process and the statistical significance of the observed paired differences was measured using the non-parametric Wilcoxon signed rank test.

5.5.2 Results and Discussion

Figure 5.4 shows the distribution of the F-measure scores obtained in 2 of our 18 data sets for the relevance threshold of 0.75. The full results for all data sets and thresholds can be found in Appendix A. For each combination of data set and regression algorithm the graphs provide 29 box-plots, one for each of the 28 mentioned sampling approaches plus the alternative of using the original data (tagged as *none* in the graphs). The box plots show the distribution of the F_1 scores of all variants of each learner on each data set. This distribution is obtained using the results from the 30 repetitions of the $3 \times 10-$fold cross validation process. These two particular data sets were chosen because they represent two different patterns of results occurring similarly through the relevance threshold range. Results on data set *a2* are among the best from the re-sampling approaches perspective. The *acceleration* data set can be regarded as an example of a domain where the advantage of re-sampling approaches is not so marked.

Although in some cases, as previously mentioned, the behaviour of the re-sampling approaches is similar for all the considered thresholds, in some domains there are considerable differences for the several threshold values. Two examples of this behaviour occur in data sets such as *a3* and *dAiler*. Figure 5.5 shows the results for the *a3* data set for each relevance threshold and each learning system. In this case, we can observe similar results for the SVM learner across all the thresholds and a remarkable improvement in the performance of the re-sampling strategies for the higher relevance thresholds in Random Forests and MARS. In Figure 5.6 we can examine the results for the *dAiler* data set. For this case, the results show substantial difference among the relevance thresholds and the learning systems. In fact, for MARS there is an improvement in the re-sampling strategies performance for the higher thresholds, the inverse is observed for the Random Forest learner, and the results for the SVM are

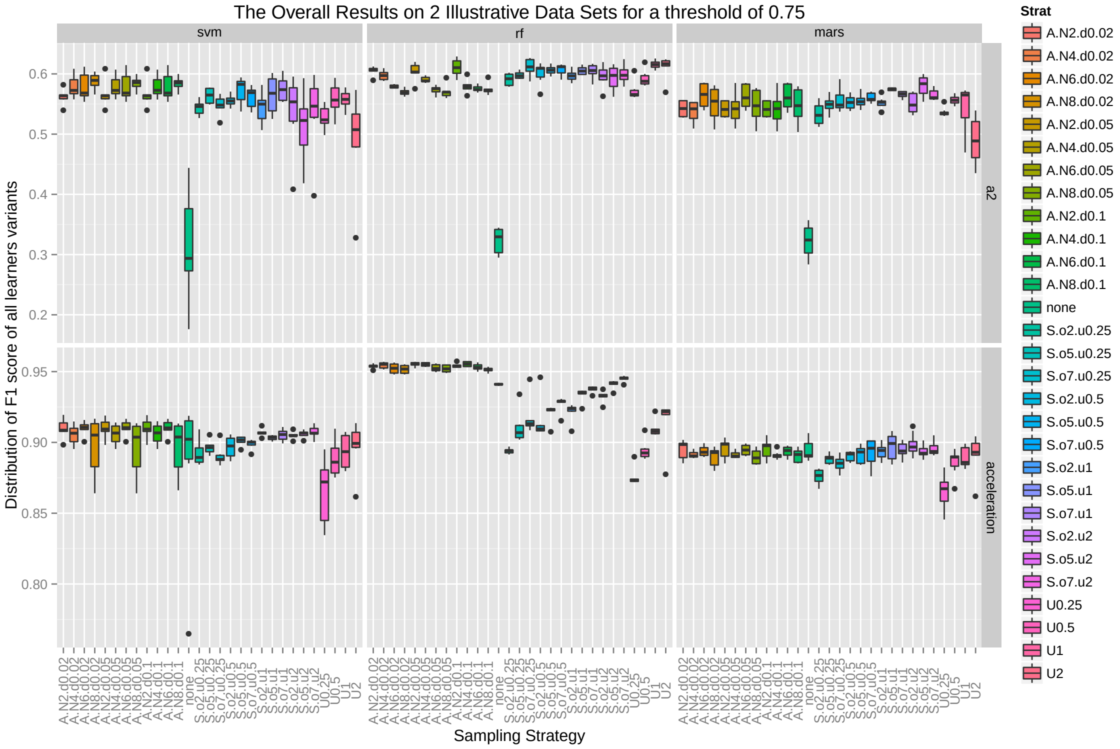


Figure 5.4: Behaviour of the re-sampling strategies on the *a2* and *acceleration* data sets with a relevance threshold of 0.75 (S - SmoteR ; U - under-sampling; A - adaptive sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

more stable across the relevance thresholds.

When taking into consideration all 18 data sets, in most cases we have an advantage of the re-sampling approaches for all the relevance thresholds considered. This can be confirmed when looking at the overall results in terms of the statistical significance of the paired differences between each sampling approach and the alternative of using the original data (the baseline). Table 5.3 summarises the results of the paired comparison of each of the 28 sampling variants against the baseline of using the given imbalanced data set for a relevance threshold of 0.75. Table 5.4 and Table 5.5 show the same results for 0.9 and 0.95 thresholds, respectively. Each sampling strategy was compared against the baseline 360 times (20 learning variants times 18 data sets). For each paired comparison we check the statistical significance of the difference in the medianF_1 score obtained with the respective sampling approach and with the baseline. These averages were estimated using a 3×10-fold CV process. We counted the number of significant wins and losses of each of the 31 sampling variants on these 360 paired comparisons using two significance levels (99% and 95%).

The results for the 0.75 relevance threshold of Table 5.3 provide clear evidence of the advantage of using re-sampling approaches when the task is to predict rare extreme values of a continuous target variable for domains with an average of 11.9% of rare cases.

In effect, we can observe an overwhelming advantage in terms of number of statistically significant wins over the alternative of using the data set as given (i.e. no re-sampling). For instance, the particular configuration of using under-sampling with 200% (*U2*) was significantly better than the alternative of using the given data set on 49.2% of the 360 considered situations, while only on 17.8% of the cases under-sampling actually lead to a significantly worse model. The remarkable performance of this very simple re-sampling strategy is even re-enforced by the fact that it uses a much smaller training set than the other alternatives, which means lower computation costs. The SmoteR variant with 700% over-sampling and 200% under-sampling (*S.o7.u2*) also achieved very good results (59.7% significant wins and 8.3% significant losses). The adaptive sampling variant with 6 bins and disturbance set to 0.02 (*A.N6.d0.02*) achieved results similar to SmoteR with 59.4% significant wins and 9.4% significant losses.

For a relevance threshold of 0.9 the results of Table 5.4 show even further advantages when compared to the results of threshold 0.75. In this case, we have a lower percentage of rare cases and yet we achieve more significant wins and less significant losses for all the sampling strategies. For instance, the under-sampling variant with

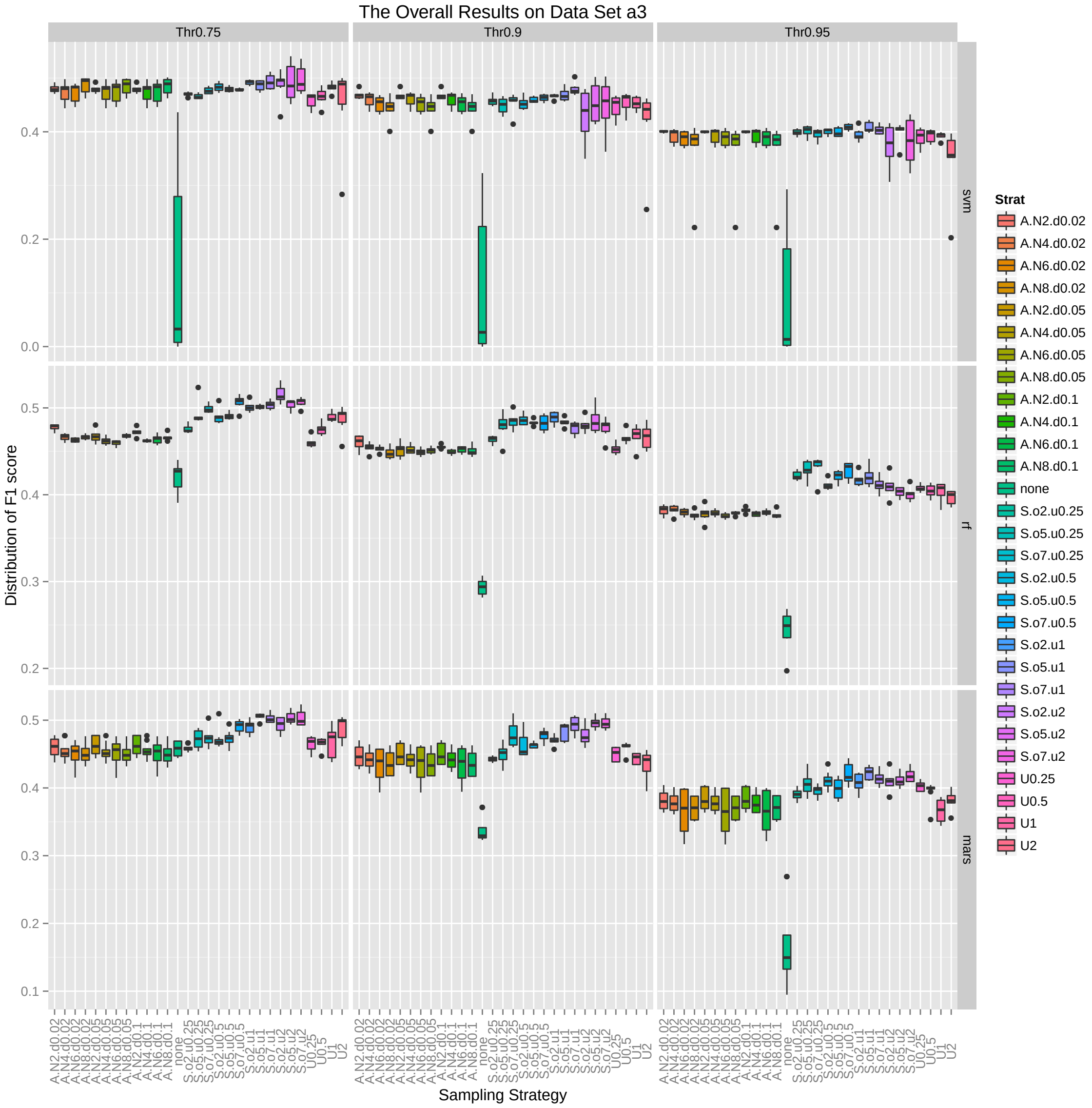


Figure 5.5: Behaviour of the re-sampling strategies on the *a3* data set across the relevance thresholds considered (S - SmoteR ; U - under-sampling; A - adaptive sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

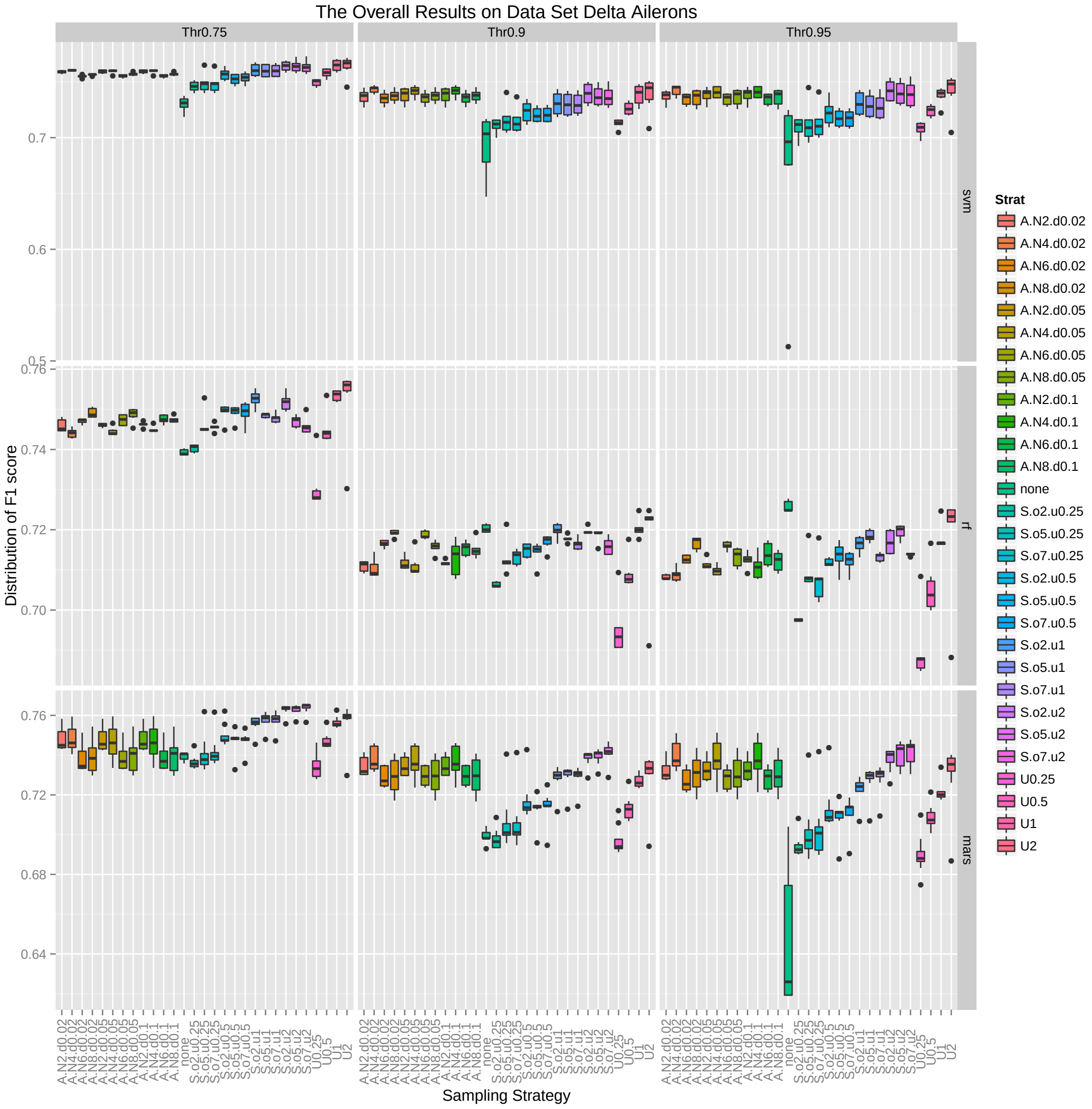


Figure 5.6: Behaviour of the re-sampling strategies on the Delta Ailerons ($dAiler$) data set across the relevance thresholds considered (S - SmoteR ; U - under-sampling; A - adaptive sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

Sampling Strat.	Win> 99%	Win > 95%	Loss> 99%	Loss> 95%	Insignif. Diff.
U0.25	99	120	115	129	111
U0.5	122	145	86	94	121
U1	161	176	69	81	103
U2	159	177	54	64	119
S.o2.u0.25	102	119	106	120	121
S.o5.u0.25	106	127	86	102	131
S.o7.u0.25	109	126	85	98	136
S.o2.u0.5	126	162	71	81	117
S.o5.u0.5	139	162	43	53	145
S.o7.u0.5	142	181	43	51	128
S.o2.u1	168	192	40	45	123
S.o5.u1	176	200	29	38	122
S.o7.u1	178	199	26	38	123
S.o2.u2	175	203	23	37	120
S.o5.u2	179	203	17	23	134
S.o7.u2	188	215	23	30	115
A.N2.d0.02	179	203	15	22	135
A.N4.d0.02	176	206	15	21	133
A.N6.d0.02	178	214	25	34	112
A.N8.d0.02	171	210	29	37	113
A.N2.d0.05	182	201	14	24	135
A.N4.d0.05	178	212	17	22	126
A.N6.d0.05	181	210	26	35	115
A.N8.d0.05	169	209	29	38	113
A.N2.d0.1	182	203	15	24	133
A.N4.d0.1	177	206	17	23	131
A.N6.d0.1	179	212	27	33	115
A.N8.d0.1	171	210	31	39	111

Table 5.3: Summary of the paired comparisons to the no sampling baseline with relevance threshold set to 0.75 (S - SmoteR ; U - under-sampling; A - adaptive sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

Sampling Strat.	Win> 99%	Win> 95%	Loss> 99%	Loss> 95%	Insignif. Diff.
U0.25	145	171	112	114	75
U0.5	166	196	85	94	70
U1	182	209	67	72	79
U2	189	214	49	59	87
S.o2.u0.25	150	176	85	98	86
S.o5.u0.25	150	184	69	81	95
S.o7.u0.25	153	183	68	74	103
S.o2.u0.5	169	203	59	65	92
S.o5.u0.5	178	207	49	61	92
S.o7.u0.5	188	214	35	50	96
S.o2.u1	195	220	44	52	88
S.o5.u1	208	224	15	28	108
S.o7.u1	212	230	17	27	103
S.o2.u2	208	238	13	20	102
S.o5.u2	229	256	13	17	87
S.o7.u2	222	254	16	21	85
A.N2.d0.02	224	259	8	12	89
A.N4.d0.02	219	253	9	18	89
A.N6.d0.02	218	255	13	17	88
A.N8.d0.02	209	256	19	21	83
A.N2.d0.05	226	257	8	14	89
A.N4.d0.05	222	252	10	18	90
A.N6.d0.05	220	254	13	17	89
A.N8.d0.05	212	250	19	20	90
A.N2.d0.1	223	262	8	13	85
A.N4.d0.1	222	246	10	15	99
A.N6.d0.1	222	252	13	17	91
A.N8.d0.1	209	249	18	21	90

Table 5.4: Summary of the paired comparisons to the no sampling baseline with relevance threshold set to 0.9 (S - SmoteR ; U - under-sampling; A - adaptive sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

Sampling Strat.	Win> 99%	Win> 95%	Loss> 99%	Loss> 95%	Insignif. Diff.
U0.25	160	182	110	113	65
U0.5	179	198	83	86	76
U1	191	210	53	64	86
U2	197	225	45	53	82
S.o2.u0.25	159	181	81	92	87
S.o5.u0.25	171	192	69	75	93
S.o7.u0.25	170	194	66	71	95
S.o2.u0.5	186	209	58	65	86
S.o5.u0.5	185	209	39	51	100
S.o7.u0.5	200	227	33	51	82
S.o2.u1	204	230	34	43	87
S.o5.u1	212	242	13	22	96
S.o7.u1	219	244	19	31	85
S.o2.u2	228	271	12	16	73
S.o5.u2	247	266	13	18	76
S.o7.u2	254	271	17	28	61
A.N2.d0.02	233	264	14	15	81
A.N4.d0.02	230	266	11	16	78
A.N6.d0.02	232	263	11	16	81
A.N8.d0.02	233	266	17	20	74
A.N2.d0.05	236	262	10	13	85
A.N4.d0.05	231	258	11	19	83
A.N6.d0.05	230	265	11	14	81
A.N8.d0.05	228	263	20	22	75
A.N2.d0.1	237	261	8	11	88
A.N4.d0.1	230	263	12	18	79
A.N6.d0.1	238	263	12	17	80
A.N8.d0.1	227	264	19	23	73

Table 5.5: Summary of the paired comparisons to the no sampling baseline with relevance threshold set to 0.95 (S - SmoteR ; U - under-sampling; A - adaptive sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

200% (*U2*) had 59.4% significant wins and 16.4% losses when compared to using the original data set. The SmoteR approach lead to 71.1% wins and 4.7% losses for the variant using 500% and 200% of over-sampling and under-sampling percentages respectively (*S.o5.u2*). As for the adaptive sampling strategy, the combination of 2 bins and 0.01 for parameter d (*A.N2.d0.01*) resulted in 72.8% of significant wins and 3.6% of significant losses.

Finally, the results for the 0.95 threshold are also remarkable. For a significance level of 95% all the sampling strategies improve the percentage of wins over the alternative of not applying re-sampling. The under-sampling variant using 200% (*U2*) has 62.5% wins and 14.7% losses; the SmoteR variant with 200% of over-sampling and 200% of under-sampling (*S.o2.u2*) presents 75.3% wins and 4.4% losses; and, finally, the Adaptive Sampling variant with 4 bins and d set to 0.02 (*A.N4.d0.02*) leads to 73.9% wins and 4.4% losses.

Table 5.6 also confirms the advantage of applying a re-sampling strategy across the 18 data sets. Similar tables for the 0.9 and 0.95 thresholds can be found in Appendix A. These results show that, for each data set, there is always a sampling strategy which allows to improve the F_1 score.

For a better understanding of the differences among the several strategies we have also conducted paired comparisons between all proposed alternatives for the different relevance thresholds considered. Figure 5.7 presents the results for the 0.75 threshold of these pairwise comparisons using the Wilcoxon signed rank test with Bonferroni correction for multiple testing. To provide a better interpretability, the test p-values were transformed into symbols according to following key: ’+’ and ’++’ signs represent win with 95% and 99% confidence respectively, ’-’ and ’–’ signs represent a loss with 95% and 99% confidence respectively, and a blank space represents that no significant difference was found in the test. The symbols should be read regarding the strategy present in the table line. For instance, a ’++’ symbol in a cell means that the strategy in the row achieves a significantly better result than the strategy on the column with 99% confidence.

In Figures 5.8 and 5.9 we present the results for the pairwise comparisons using Wilcoxon signed rank test with Bonferroni correction for the 0.9 and 0.95 relevance thresholds, respectively.

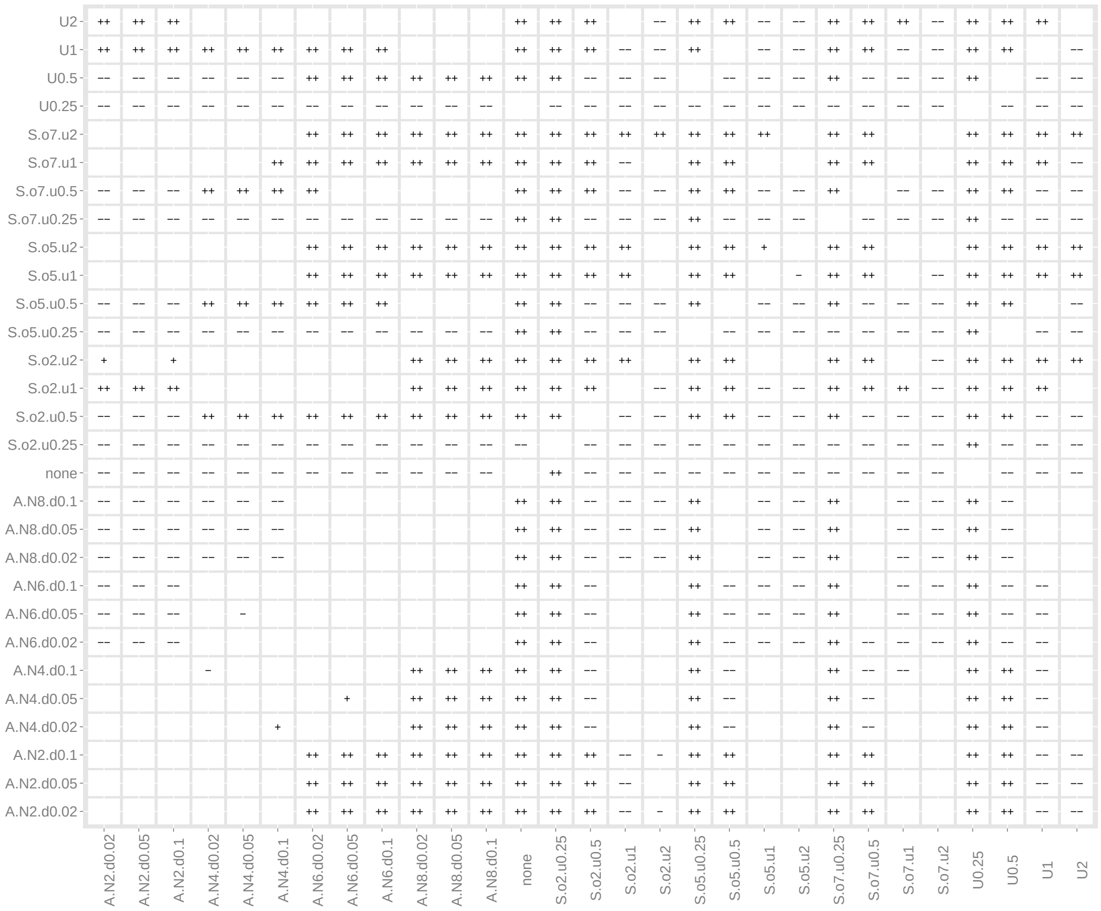
These results confirm that, for all the relevance thresholds, using the data set as given, i.e. applying no re-sampling strategy, always represents a loss with 99% confidence with the exception for the 0.75 threshold of the *S.o2.u0.25* strategy and the *U0.25*

Data set	none	Under-sampling	SmoteR	Adaptive Sampling
a1	0.4634805	0.7027627	**0.712006**	0.7004192
a2	0.3197353	0.5731104	**0.5812283**	0.5733544
a3	0.3522209	0.482336	**0.5024598**	0.4711112
a4	0.4296375	0.577266	0.5921964	**0.5952717**
a5	0.1585692	0.4793509	**0.5024495**	0.4983746
a6	0.3802069	**0.5042783**	0.5038622	0.4885777
a7	0.2333013	0.3585283	**0.3739525**	0.3631748
Abalone	0.7161421	0.7325742	0.733171	**0.7342574**
Accel	0.9044244	0.9063044	0.9142926	**0.9174297**
dAiler	0.7365465	**0.7606014**	0.7601443	0.750816
dElev	0.7300003	0.7493701	0.7485255	**0.7501716**
availPwr	0.9295915	0.9283252	**0.9330286**	0.9222684
bank8FM	0.9455891	0.946326	0.9466964	**0.9514199**
boston	0.898961	0.8941507	**0.901301**	0.89677
cpuSm	0.2597305	0.2800133	0.2913918	**0.3056211**
fuelCons	0.8967333	0.8958695	**0.9047295**	0.8980186
maximalTorque	0.9649971	0.9711579	0.9702372	**0.9834484**
heat	0.9356263	0.9421861	0.9462077	**0.9631721**

Table 5.6: Best mean F_1 score of each sampling approach for all learning systems with a relevance threshold set to 0.75

approach. We highlight the poor performance of under-sampling at {25, 50}% for all the relevance thresholds. In effect, these strategies almost always loose against the other strategies. There is also a similar behaviour of the SmoteR strategies with the same under-sampling percentages. The adaptive sampling strategy with 8 bins shows a poor performance across all thresholds only presenting advantages (wins) when compared with under-sampling or SmoteR both with under-sampling percentages of at {25, 50}% . We must emphasise as well the lack of statistical significance among the differences of the Adaptive Sampling strategies with 2 and 4 bins for all thresholds. For the two higher values of the relevance threshold, for nearly all the SmoteR strategies with under-sampling at {100, 200}% and the Adaptive Sampling strategies with 4 and 6 bins the existing differences are not statistically significant.

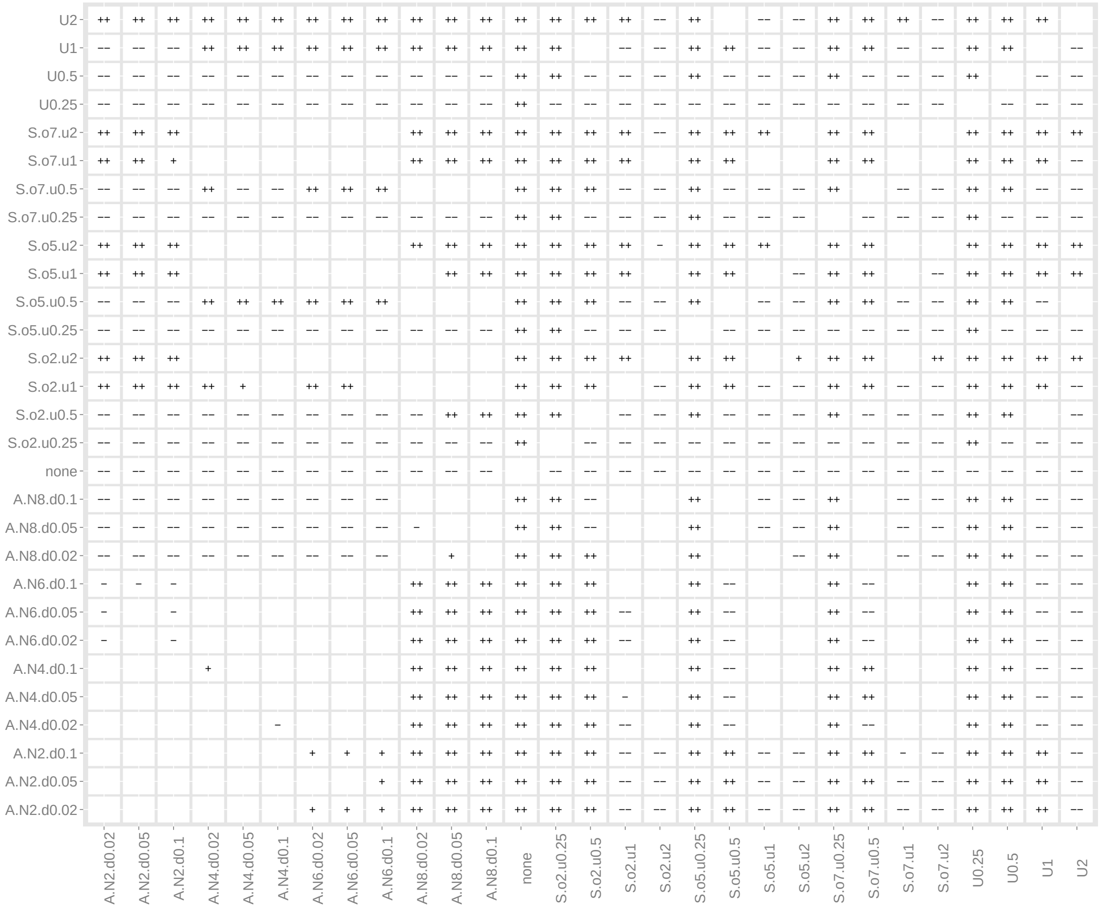
In summary, the results of our experimental comparisons provide clear evidence on the validity of the re-sampling approaches we have proposed. The overall best results are obtained with under-sampling with {100, 200}%, SmoteR with the same under-sampling percentages, and adaptive sampling with the number of bins ranging from 2 to 6. We should stress that adaptive sampling is easier to use from the point of view of the user as it requires settings a smaller number of parameters.We highlight that



	A.N2.d0.02	A.N2.d0.05	A.N2.d0.1	A.N4.d0.02	A.N4.d0.05	A.N4.d0.1	A.N6.d0.02	A.N6.d0.05	A.N6.d0.1	A.N8.d0.02	A.N8.d0.05	A.N8.d0.1	none	S.o2.u0.25	S.o2.u0.5	S.o2.u1	S.o2.u2	S.o5.u0.25	S.o5.u0.5	S.o5.u1	S.o5.u2	S.o7.u0.25	S.o7.u0.5	S.o7.u1	S.o7.u2	U0.25	U0.5	U1	U2
U2	++	++	++										++	++	++		−−	++	++	−−	−−	++	++	++	−−	++	++	++	
U1	++	++	++	++	++	++	++	++	++				++	++	++	−−	−−	++		−−	−−	++	++	−−	−−	++	++		−−
U0.5	−−	−−	−−	−−	−−	−−	++	++	++	++	++	++	++	++	−−	−−	−−		−−	−−	−−	++	−−	−−	−−	++		−−	−−
U0.25	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−		−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−		−−	−−	−−
S.o7.u2							++	++	++	++	++	++	++	++	++	++	++	++	++	++		++	++			++	++	++	++
S.o7.u1						++	++	++	++	++	++	++	++	++	++	−−		++	++			++	++			++	++	++	−−
S.o7.u0.5	−−	−−	−−	++	++	++	++						++	++	++	−−	−−	++	++	−−	−−	++		−−	−−	++	++	−−	−−
S.o7.u0.25	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	++	++	−−	−−	−−	++	−−	−−	−−		−−	−−	−−	++	−−	−−	−−
S.o5.u2							++	++	++	++	++	++	++	++	++	++		++	++	+		++	++			++	++	++	++
S.o5.u1							++	++	++	++	++	++	++	++	++	++		++	++		−	++	++		−−	++	++	++	++
S.o5.u0.5	−−	−−	−−	++	++	++	++	++	++				++	++	−−	−−	−−	++		−−	−−	++	−−	−−	−−	++	++		−−
S.o5.u0.25	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	++	++	−−	−−	−−		−−	−−	−−	−−	−−	−−	−−	++		−−	−−
S.o2.u2	+		+							++	++	++	++	++	++	++		++	++			++	++		−−	++	++	++	++
S.o2.u1	++	++	++							++	++	++	++	++	++		−−	++	++	−−	−−	++	++	++	−−	++	++	++	
S.o2.u0.5	−−	−−	−−	++	++	++	++	++	++	++	++	++	++	++		−−	−−	++	++	−−	−−	++	−−	−−	−−	++	++	−−	−−
S.o2.u0.25	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−		−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	++	−−	−−	−−
none	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−		++	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−	−−		−−	−−	−−
A.N8.d0.1	−−	−−	−−	−−	−−	−−							++	++	−−	−−	−−	++		−−	−−	++		−−	−−	++	−−		
A.N8.d0.05	−−	−−	−−	−−	−−	−−							++	++	−−	−−	−−	++		−−	−−	++		−−	−−	++	−−		
A.N8.d0.02	−−	−−	−−	−−	−−	−−							++	++	−−	−−	−−	++		−−	−−	++		−−	−−	++	−−		
A.N6.d0.1	−−	−−	−−										++	++	−−			++	−−	−−	−−	++		−−	−−	++	−−	−−	
A.N6.d0.05	−−	−−	−−		−								++	++	−−			++	−−	−−	−−	++		−−	−−	++	−−	−−	
A.N6.d0.02	−−	−−	−−										++	++	−−			++	−−	−−	−−	++	−−	−−	−−	++	−−	−−	
A.N4.d0.1				−						++	++	++	++	++	−−			++	−−			++	−−	−−		++	++	−−	
A.N4.d0.05								+		++	++	++	++	++	−−			++	−−			++	−−			++	++	−−	
A.N4.d0.02						+				++	++	++	++	++	−−			++	−−			++	−−			++	++	−−	
A.N2.d0.1							++	++	++	++	++	++	++	++	++	−−	−	++	++			++	++			++	++	−−	−−
A.N2.d0.05							++	++	++	++	++	++	++	++	++	−−		++	++			++	++			++	++	−−	−−
A.N2.d0.02							++	++	++	++	++	++	++	++	++	−−	−	++	++			++	++			++	++	−−	−−

Figure 5.7: Pairwise comparisons with Wilcoxon signed rank test of all strategies against each other with Bonferroni correction for a relevance threshold of 0.75 (S - SmoteR ; U - under-sampling; A - adaptive sampling; ox - x × 100% over-sampling; ux - x × 100% under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

random under-sampling uses a much smaller training set than the other alternative approaches which means lower computational costs. This does not happen with SmoteR which is a more complex algorithm, with a possibly very large training set and internally requiring the evaluation of distances for determining the k nearest neighbours. Adaptive sampling is a more stable method since it keeps the training set size relatively unchanged. Also, the good performance of this approach is re-enforced by the reduced computational costs when compared with SmoteR strategy and the user friendly aspect associated with the setting of less parameters.



	A.N2.d0.02	A.N2.d0.05	A.N2.d0.1	A.N4.d0.02	A.N4.d0.05	A.N4.d0.1	A.N6.d0.02	A.N6.d0.05	A.N6.d0.1	A.N8.d0.02	A.N8.d0.05	A.N8.d0.1	none	S.o2.u0.25	S.o2.u0.5	S.o2.u1	S.o2.u2	S.o5.u0.25	S.o5.u0.5	S.o5.u1	S.o5.u2	S.o7.u0.25	S.o7.u0.5	S.o7.u1	S.o7.u2	U0.25	U0.5	U1	U2
U2	++	++	++	++	++	++	++	++	++	++	++	++	++	++	++	++	--	++		--	--	++	++	++	--	++	++	++	
U1	--	--	--	++	++	++	++	++	++	++	++	++	++	++		--	--	++	++	--	--	++	++	--	--	++	++		--
U0.5	--	--	--	--	--	--	--	--	--	--	--	--	++	++	--	--	--	++	--	--	--	++	--	--	--	++		--	--
U0.25	--	--	--	--	--	--	--	--	--	--	--	--	++	--	--	--	--	--	--	--	--	--	--	--	--		--	--	--
S.o7.u2	++	++	++							++	++	++	++	++	++	++	--	++	++	++		++	++			++	++	++	++
S.o7.u1	++	++	+							++	++	++	++	++	++	++		++	++			++	++			++	++	++	--
S.o7.u0.5	--	--	--	++	--	--	++	++	++				++	++	++	--	--	++	--	--	--	++		--	--	++	++	--	--
S.o7.u0.25	--	--	--	--	--	--	--	--	--	--	--	--	++	++	--	--	--	++	--	--	--		--	--	--	++	--	--	--
S.o5.u2	++	++	++							++	++	++	++	++	++	++	-	++	++	++		++	++			++	++	++	++
S.o5.u1	++	++	++								++	++	++	++	++	++		++	++		--	++	++		--	++	++	++	++
S.o5.u0.5	--	--	--	++	++	++	++	++	++				++	++	++	--	--	++		--	--	++	++	--	--	++	++	--	
S.o5.u0.25	--	--	--	--	--	--	--	--	--	--	--	--	++	++	--	--	--		--	--	--	--	--	--	--	++	--	--	--
S.o2.u2	++	++	++										++	++	++	++		++	++		+	++	++		++	++	++	++	++
S.o2.u1	++	++	++	++	+		++	++					++	++	++		--	++	++	--	--	++	++	--	--	++	++	++	--
S.o2.u0.5	--	--	--	--	--	--	--	--	--	--	++	++	++	++		--	--	++	--	--	--	++	--	--	--	++	++		--
S.o2.u0.25	--	--	--	--	--	--	--	--	--	--	--	--	++		--	--	--	--	--	--	--	--	--	--	--	++	--	--	--
none	--	--	--	--	--	--	--	--	--	--	--	--		--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
A.N8.d0.1	--	--	--	--	--	--	--	--	--				++	++	--			++		--	--	++		--	--	++	++	--	--
A.N8.d0.05	--	--	--	--	--	--	--	--	--	-			++	++	--			++		--	--	++		--	--	++	++	--	--
A.N8.d0.02	--	--	--	--	--	--	--	--	--		+		++	++	++			++			--	++		--	--	++	++	--	--
A.N6.d0.1	-	-	-							++	++	++	++	++	++			++	--			++	--			++	++	--	--
A.N6.d0.05	-		-							++	++	++	++	++	++	--		++	--			++	--			++	++	--	--
A.N6.d0.02	-		-							++	++	++	++	++	++	--		++	--			++	--			++	++	--	--
A.N4.d0.1				+						++	++	++	++	++	++			++	--			++	++			++	++	--	--
A.N4.d0.05										++	++	++	++	++	++	-		++	--			++	++			++	++	--	--
A.N4.d0.02						-				++	++	++	++	++	++	--		++	--			++	--			++	++	--	--
A.N2.d0.1							+	+	+	++	++	++	++	++	++	--	--	++	++	--	--	++	++	-	--	++	++	++	--
A.N2.d0.05									+	++	++	++	++	++	++	--	--	++	++	--	--	++	++	--	--	++	++	++	--
A.N2.d0.02							+	+	+	++	++	++	++	++	++	--	--	++	++	--	--	++	++	--	--	++	++	++	--

Figure 5.8: Pairwise comparisons with Wilcoxon signed rank test of all strategies against each other with Bonferroni correction for a relevance threshold of 0.9 (S - SmoteR ; U - under-sampling; A - adaptive sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

	A.N2.d0.02	A.N2.d0.05	A.N2.d0.1	A.N4.d0.02	A.N4.d0.05	A.N4.d0.1	A.N6.d0.02	A.N6.d0.05	A.N6.d0.1	A.N8.d0.02	A.N8.d0.05	A.N8.d0.1	none	S.o2.u0.25	S.o2.u0.5	S.o2.u1	S.o2.u2	S.o5.u0.25	S.o5.u0.5	S.o5.u1	S.o5.u2	S.o7.u0.25	S.o7.u0.5	S.o7.u1	S.o7.u2	U0.25	U0.5	U1	U2
U2	++	++	++	++	++	++	++	++	++	++	++	++	++	++	++	++	--	++		--	--	++	++	++	--	++	++	++	
U1	--	--	--	++	++	++	++	++	++	++	++	++	++	++		--	--	++	++	--	--	++	++	--	--	++	++		--
U0.5	--	--	--	--	--	--	--	--	--	--	--	--	++	++	--	--	--	++	--	--	--	++	--	--	--	++		--	--
U0.25	--	--	--	--	--	--	--	--	--	--	--	--	++	--	--	--	--	--	--	--	--	--	--	--	--		--	--	--
S.o7.u2	++	++	++							++	++	++	++	++	++	++		++	++			++	++			++	++	++	++
S.o7.u1	++	++	++							++	++	++	++	++	++	--		++	++			++	++			++	++	++	--
S.o7.u0.5	--	--	--	--	--	--							++	++	++	--	--	++	--	--	--	++		--	--	++	++	--	--
S.o7.u0.25	--	--	--	--	--	--	--	--	--	--	--	--	++	++	--	--	--	--	--	--	--		--	--	--	++	--	--	--
S.o5.u2	++	++	++							++	++	++	++	++	++	++	-	++	++			++	++			++	++	++	++
S.o5.u1	++	++	++							++	++	++	++	++	++	++		++	++			++	++			++	++	++	++
S.o5.u0.5	--	--	--	--	--	--	--	--	--				++	++	++	--	--	++		--	--	++	++	--	--	++	++	--	
S.o5.u0.25	--	--	--	--	--	--	--	--	--	--	--	--	++	++	--	--	--		--	--	--	++	--	--	--	++	--	--	--
S.o2.u2	++	++	++								+	+	++	++	++	++		++	++		+	++	++			++	++	++	++
S.o2.u1	++	++	++							+	++	+	++	++	++		--	++	++	--	--	++	++	++	--	++	++	++	--
S.o2.u0.5	--	--	--	--	--	--	--	--	--	--	--	--	++	++		--	--	++	--	--	--	++	--	--	--	++	++		--
S.o2.u0.25	--	--	--	--	--	--	--	--	--	--	--	--	++		--	--	--	--	--	--	--	--	--	--	--	++	--	--	--
none	--	--	--	--	--	--	--	--	--	--	--	--		--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
A.N8.d0.1	--	--	--	--	--	--	--	--	--				++	++	++	-	-	++		--	--	++		--	--	++	++	--	--
A.N8.d0.05	--	--	--	--	--	--	--	--	--				++	++	++	--	-	++		--	--	++		--	--	++	++	--	--
A.N8.d0.02	--	--	--	--	--	--	--	--	--				++	++	++	-		++		--	--	++		--	--	++	++	--	--
A.N6.d0.1	--	--	--	--	-	--				++	++	++	++	++	++			++	++			++				++	++	--	--
A.N6.d0.05	--	--	--	--						++	++	++	++	++	++			++	++			++				++	++	--	--
A.N6.d0.02	--	--	--	-						++	++	++	++	++	++			++	++			++				++	++	--	--
A.N4.d0.1									++	++	++	++	++	++	++			++	++			++	++			++	++	--	--
A.N4.d0.05									+	++	++	++	++	++	++			++	++			++	++			++	++	--	--
A.N4.d0.02							+	++	++	++	++	++	++	++	++			++	++			++	++			++	++	--	--
A.N2.d0.1							++	++	++	++	++	++	++	++	++	--	--	++	++	--	--	++	++	--	--	++	++	++	--
A.N2.d0.05							++	++	++	++	++	++	++	++	++	--	--	++	++	--	--	++	++	--	--	++	++	++	--
A.N2.d0.02							++	++	++	++	++	++	++	++	++	--	--	++	++	--	--	++	++	--	--	++	++	++	--

Figure 5.9: Pairwise comparisons with Wilcoxon signed rank test of all strategies against each other with Bonferroni correction for a relevance threshold of 0.95 (S - SmoteR ; U - under-sampling; A - adaptive sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

Chapter 6

Conclusion

6.1 Summary

The problem of forecasting rare values of a nominal target variable, usually known as the problem of class imbalance, has been extensively studied. For regression tasks, when the target variable is continuous, and despite of the existence of several important real world applications, few works exist on forecasting rare and extreme values.

In this thesis we have addressed the problem of imbalanced domains for regression tasks. We have provided an extensive survey on the existing performance assessment metrics and strategies for this problem and have presented three new re-sampling approaches to tackle such tasks. The main goals of our study were to: i) highlight the importance of considering adequate metrics for this problem type; ii) present the state of the art on performance assessment metrics and approaches for imbalanced data sets; iii) provide as extensive survey of the existing methods to tackle the problem of imbalanced domains for classification and regression tasks; and iv) propose and perform an experimental analysis of three new re-sampling approaches for regression tasks under imbalanced domains.

Through an extensive set of experiments carried out on a diverse set of problems and using rather different learning algorithms, we have shown the competitiveness of our proposals. The key advantages of these re-sampling methods are their simplicity and versatility. These strategies are data pre-processing methods which simply manipulate the distribution of the available training data thus allowing the use of any standard regression tool on these particular prediction tasks.

In most cases, the re-sampling approaches proposed present an advantage for all the relevance thresholds considered. Moreover, the higher the relevance threshold used the higher the impact of the re-sampling strategies. Regarding the random under-sampling method the good predictive performance is accompanied by lower computation costs due to the reduced size of the training set. The innovative aspects of SmoteR are confirmed by the good performance of this approach. Finally, the adaptive sampling algorithm combines a competitive performance with reduced computational costs when compared to SmoteR and an user friendly aspect because it requires setting a small number of parameters.

6.2 Future Research Directions

The prediction of rare and extreme values for continuous target variables is a scarcely studied problem. Very few attention has been given to this particular issue and, therefore, much remains to explore. Being a poorly investigated subject, a wide space for improvements exists.

In particular, other pre-processing methods already existing for classification tasks could also be adapted to these regression tasks. In effect, many existing techniques for the class imbalance problem were developed for improving former existing strategies. This could also be explored for regression.

Besides pre-processing methods, further categories of approaches could also be investigated, as the algorithms modifications, the predictions post-processing, or combinations of strategies.

The proposals we presented were only tested for a special case of the problem which associates rarity to extreme target variable values. Although this is frequent in real world applications, it would be interesting to extend our proposals into a more general framework were rarity could occur anywhere in the target variable domain.

Another interesting direction would be to investigate relations between the data set characteristics and the re-sampling strategies in order to build a meta-learner which could recommend a specific set of strategies and parameters for a given domain.

Appendix A

Experimental Results

In this annex we present the detailed experimental results on the 18 data sets.

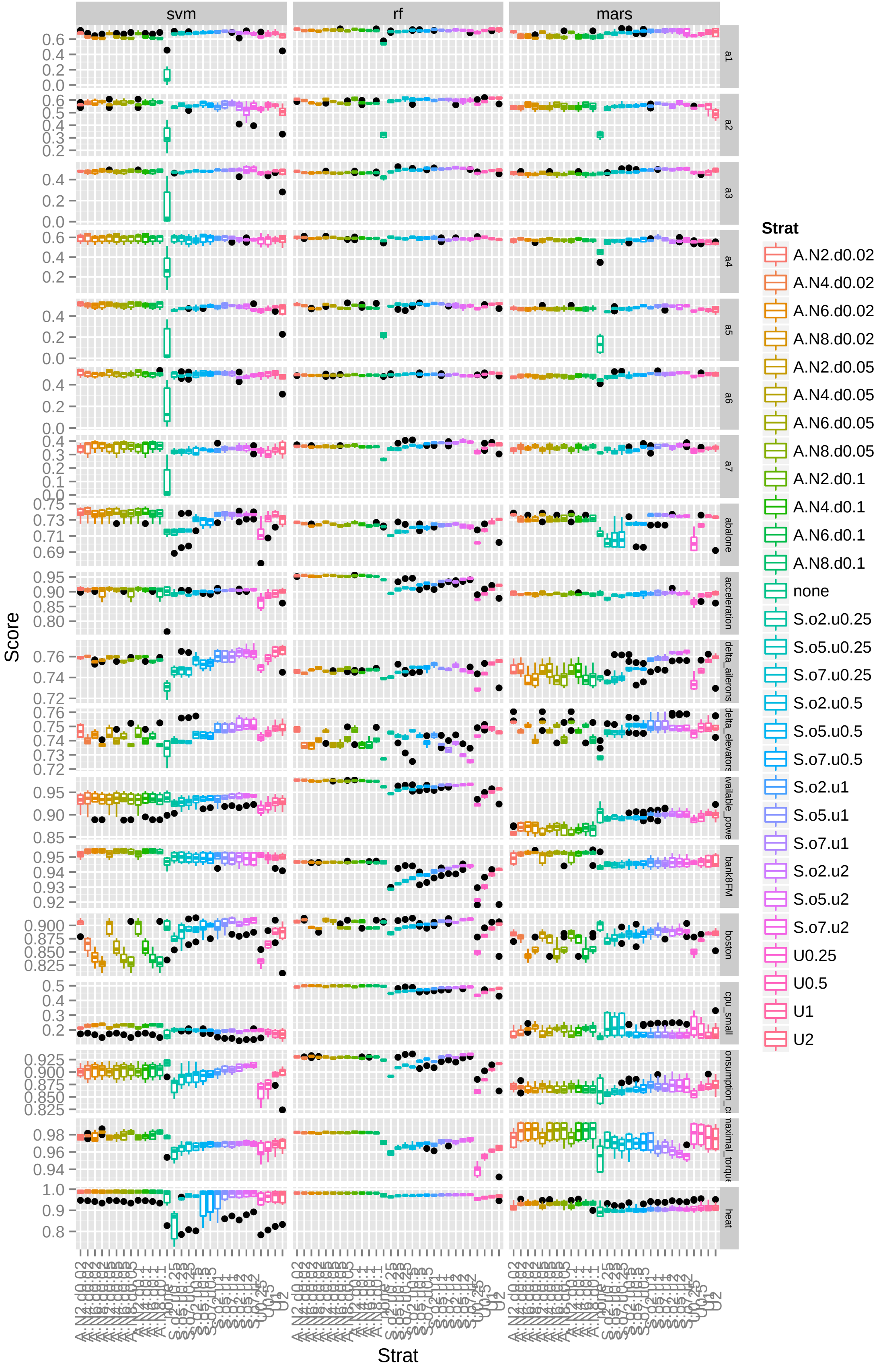


Figure A.1: Behaviour of the re-sampling strategies on 18 data sets with a relevance threshold of 0.75 (S - SmoteR ; U - under-sampling; A - Adaptive Sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

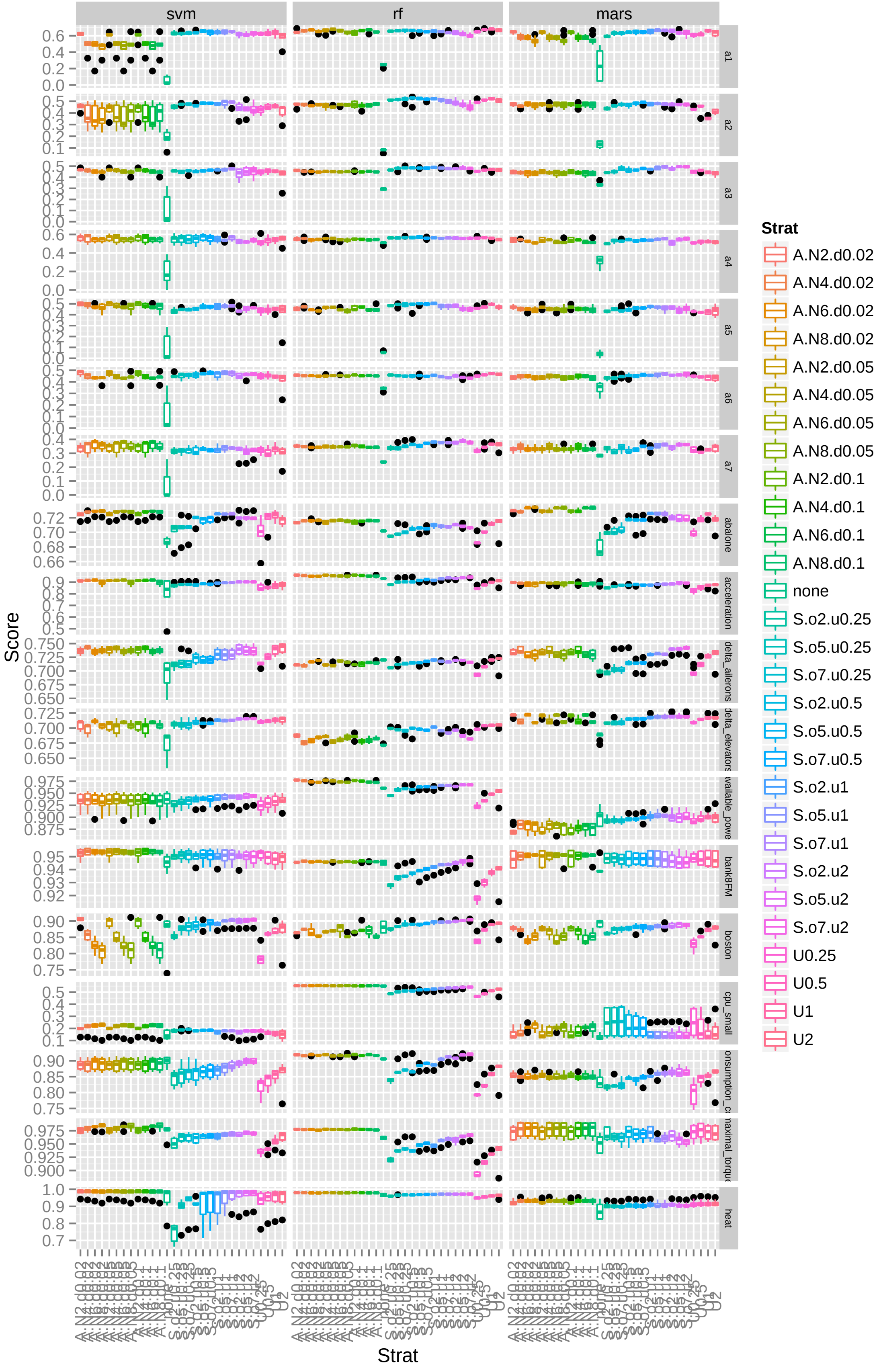


Figure A.2: Behaviour of the re-sampling strategies on 18 data sets with a relevance threshold of 0.9 (S - SmoteR ; U - under-sampling; A - Adaptive Sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

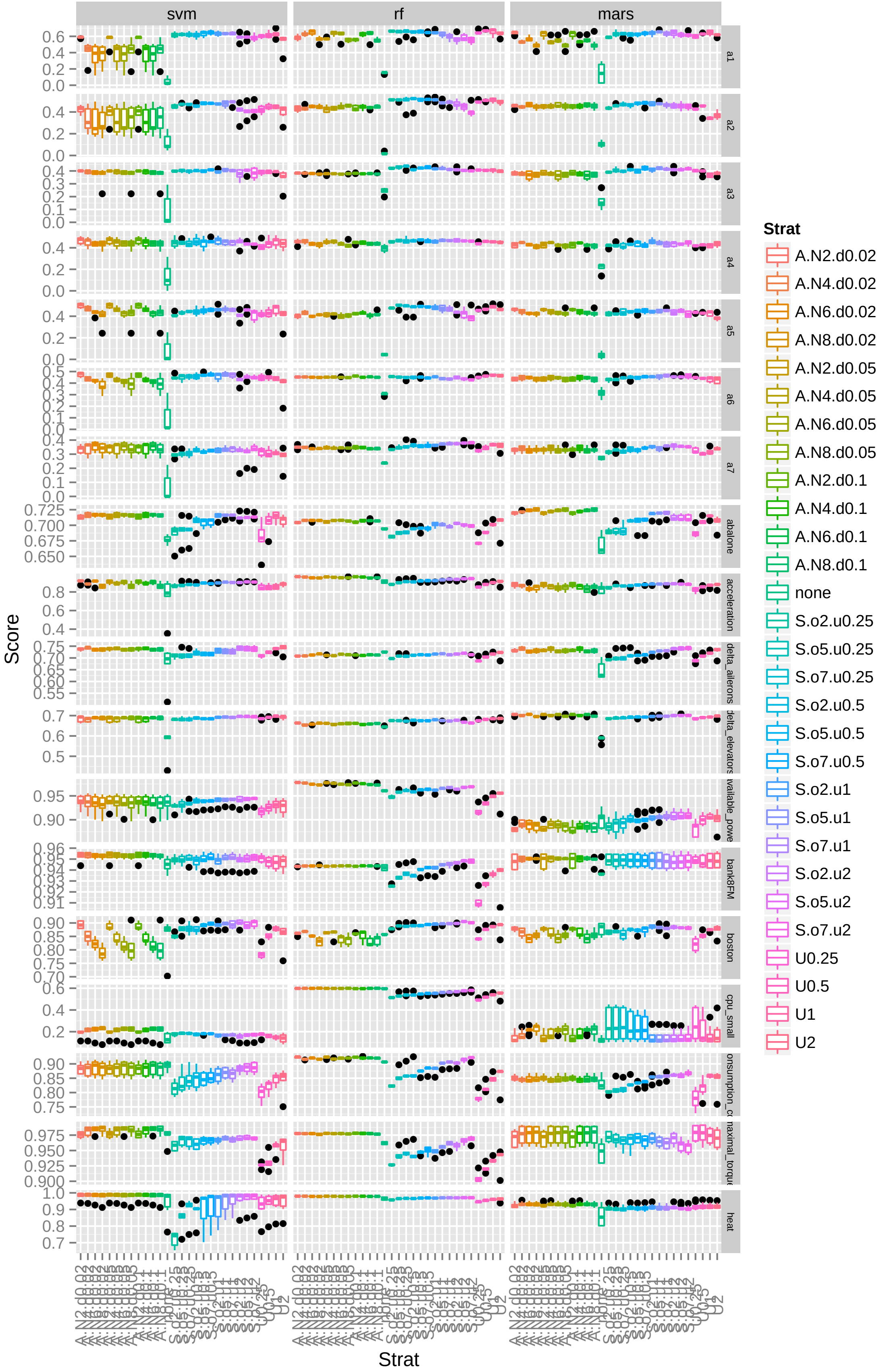


Figure A.3: Behaviour of the re-sampling strategies on 18 data sets with a relevance threshold of 0.95 (S - SmoteR ; U - under-sampling; A - Adaptive Sampling; ox - $x \times 100\%$ over-sampling; ux - $x \times 100\%$ under-sampling; Nx - nr. of intervals; dx - amount of disturbance).

Data set	none	Under-sampling	SmoteR	Adaptive
a1	0.1874861	**0.6623308**	0.6537593	0.6404921
a2	0.131762	0.4707454	**0.4946997**	0.468004
a3	0.2566088	0.4589897	**0.4850106**	0.4575761
a4	0.3327811	0.5390838	**0.5602863**	0.5536516
a5	0.06200563	0.4573975	**0.4833102**	0.4767628
a6	0.2764081	0.4524592	**0.4625432**	0.4556321
a7	0.206242	0.3391856	**0.3615381**	0.346755
Abalone	0.6884877	0.7208114	0.7210465	**0.7268933**
Accel	0.8717274	0.8906104	0.9113046	**0.9206469**
dAiler	0.7036236	0.7330744	**0.734134**	0.7317991
dElev	0.6780333	**0.7130781**	0.7120496	0.7062695
availPwr	0.9282321	0.9285287	**0.9345717**	0.9275294
bank8FM	0.9434285	0.9461111	0.9475145	**0.9504968**
boston	0.8818092	0.8866217	**0.896766**	0.8822371
cpuSm	0.2618313	0.2901348	0.3131996	**0.3183838**
fuelCons	0.875271	0.8722404	**0.8921911**	0.8844499
maximalTorque	0.9604649	0.9610309	0.9638248	**0.9790385**
heat	0.9215252	0.9407887	0.9436688	**0.962168**

Table A.1: Best mean F_1 score of each sampling approach for all learning systems with a relevance threshold set to 0.9

Data set	none	Under-sampling	SmoteR	Adaptive
a1	0.1182979	**0.6489449**	0.6421397	0.608243
a2	0.08458176	0.4678831	**0.4889491**	0.4398707
a3	0.1670689	0.3997017	**0.4213191**	0.3880046
a4	0.2432986	0.4448187	**0.4649349**	0.4526847
a5	0.05080295	0.4580971	**0.4666689**	0.4634314
a6	0.2463805	0.4502021	**0.4588278**	0.4540215
a7	0.1992734	0.331152	**0.3560618**	0.3418788
Abalone	0.6785448	0.7113216	0.713412	**0.7174341**
Accel	0.831319	0.8937601	0.9099074	**0.9146342**
dAiler	0.6780727	0.734987	**0.7352013**	0.7315092
dElev	0.5980799	**0.6933591**	0.692538	0.6870655
availPwr	0.9284757	0.9293882	**0.9373288**	0.9305915
bank8FM	0.941489	0.9457626	0.9487812	**0.9493337**
boston	0.8634719	0.8822592	**0.8941955**	0.8782996
cpuSm	0.2711127	0.3047852	0.321876	**0.3353316**
fuelCons	0.8728995	0.861138	**0.8915357**	0.8820248
maximalTorque	0.9594628	0.95959	0.9665527	**0.9787481**
heat	0.9145503	0.9403284	0.9434054	**0.9618441**

Table A.2: Best mean F_1 score of each sampling approach for all learning systems with a relevance threshold set to 0.95

References

Akbani, R., Kwek, S., and Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. In *Machine Learning: ECML 2004*, pages 39–50. Springer.

Alejo, R., García, V., Sotoca, J. M., Mollineda, R. A., and Sánchez, J. S. (2007). Improving the performance of the rbf neural networks trained with imbalanced samples. In *Computational and Ambient Intelligence*, pages 162–169. Springer.

Alejo, R., Valdovinos, R. M., García, V., and Pacheco-Sanchez, J. (2013). A hybrid method to face class overlap and class imbalance on neural networks and multi-class scenarios. *Pattern Recognition Letters*, 34(4):380–388.

Alejo Eleuterio, R., Martínez Sotoca, J., García Jiménez, V., and Valdovinos Rosas, R. M. (2011). Back propagation with balanced mse cost function and nearest neighbor editing for handling class overlap and class imbalance.

An, G. (1996). The effects of adding noise during backpropagation training on a generalization performance. *Neural Computation*, 8(3):643–674.

Bansal, G., Sinha, A. P., and Zhao, H. (2008). Tuning data mining methods for cost-sensitive regression: a study in loan charge-off forecasting. *Journal of Management Information Systems*, 25(3):315–336.

Barandela, R., Sánchez, J. S., Garcıa, V., and Rangel, E. (2003). Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851.

Barros de Almeida, M., de Pádua Braga, A., and Braga, J. P. (2000). Svm-km: speeding svms learning with a priori cluster selection and k-means. In *Neural Networks, 2000. Proceedings. Sixth Brazilian Symposium on*, pages 162–167. IEEE.

Barua, S., Islam, M., Yao, X., and Murase, K. (2012). Mwmote-majority weighted minority oversampling technique for imbalanced data set learning.

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

Batuwita, R. and Palade, V. (2009). A new performance measure for class imbalance learning. application to bioinformatics problems. In *Machine Learning and Applications, 2009. ICMLA'09. International Conference on*, pages 545–550. IEEE.

Batuwita, R. and Palade, V. (2010a). Efficient resampling methods for training support vector machines with imbalanced datasets. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1–8. IEEE.

Batuwita, R. and Palade, V. (2010b). Fsvm-cil: fuzzy support vector machines for class imbalance learning. *Fuzzy Systems, IEEE Transactions on*, 18(3):558–571.

Batuwita, R. and Palade, V. (2012). Adjusted geometric-mean: a novel performance measure for imbalanced bioinformatics datasets learning. *Journal of Bioinformatics and Computational Biology*, 10(04).

Bellinger, C., Sharma, S., and Japkowicz, N. (2012). One-class versus binary classification: Which and when? In *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, volume 2, pages 102–106. IEEE.

Bi, J. and Bennett, K. P. (2003). Regression error characteristic curves. In *Proc. of the 20th Int. Conf. on Machine Learning*, pages 43–50.

Błaszczyński, J., Deckert, M., Stefanowski, J., and Wilk, S. (2010). Integrating selective pre-processing of imbalanced data with ivotes ensemble. In *Rough Sets and Current Trends in Computing*, pages 148–157. Springer.

Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). Classification and regression trees. wadsworth & brooks. *Monterey, CA*.

Bunkhumpornpat, C., Sinapiromsaran, K., and Lursinsap, C. (2009). Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Advances in Knowledge Discovery and Data Mining*, pages 475–482. Springer.

Bunkhumpornpat, C., Sinapiromsaran, K., and Lursinsap, C. (2011). Mute: Majority under-sampling technique. In *Information, Communications and Signal Processing (ICICS) 2011 8th International Conference on*, pages 1–4. IEEE.

Bunkhumpornpat, C., Sinapiromsaran, K., and Lursinsap, C. (2012). Dbsmote: Density-based synthetic minority over-sampling technique. *Applied Intelligence*, 36(3):664–684.

Bunkhumpornpat, C. and Subpaiboonkit, S. (2013). Safe level graph for synthetic minority over-sampling techniques. In *Communications and Information Technologies (ISCIT), 2013 13th International Symposium on*, pages 570–575. IEEE.

Cain, M. and Janssen, C. (1995). Real estate price prediction under asymmetric loss. *Annals of the Institute of Statistical Mathematics*, 47(3):401–414.

Cao, P., Zhao, D., and Zaïane, O. R. (2013). A pso-based cost-sensitive neural network for imbalanced data classification. In *Trends and Applications in Knowledge Discovery and Data Mining*, pages 452–463. Springer.

Castro, C. L. and de Pádua Braga, A. (2013). Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data. *IEEE Trans. Neural Netw. Learning Syst.*, 24(6):888–899.

Chan, P. K. and Stolfo, S. J. (1998). Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *KDD*, volume 1998, pages 164–168.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

Chang, E. Y., Li, B., Wu, G., and Goh, K. (2003). Statistical learning for effective visual information retrieval. In *ICIP (3)*, pages 609–612.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *JAIR*, 16:321–357.

Chawla, N. V., Cieslak, D. A., Hall, L. O., and Joshi, A. (2008). Automatically countering imbalance and its empirical relationship to cost. *Data Mining and Knowledge Discovery*, 17(2):225–252.

Chawla, N. V., Hall, L. O., and Joshi, A. (2005). Wrapper-based computation and evaluation of sampling methods for imbalanced datasets. In *Proceedings of the 1st international workshop on Utility-based data mining*, pages 24–33. ACM.

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

Chawla, N. V., Lazarevic, A., Hall, L. O., and Bowyer, K. W. (2003). Smoteboost: Improving prediction of the minority class in boosting. In *Knowledge Discovery in Databases: PKDD 2003*, pages 107–119. Springer.

Chen, C., Liaw, A., and Breiman, L. (2004). Using random forest to learn imbalanced data. *University of California, Berkeley*.

Chen, S., He, H., and Garcia, E. A. (2010). Ramoboost: Ranked minority oversampling in boosting. *Neural Networks, IEEE Transactions on*, 21(10):1624–1642.

Chen, X.-w. and Wasikowski, M. (2008). Fast: a roc-based feature selection metric for small samples and imbalanced data classification problems. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 124–132. ACM.

Christoffersen, P. F. and Diebold, F. X. (1996). Further results on forecasting and model selection under asymmetric loss. *Journal of applied econometrics*, 11(5):561–571.

Christoffersen, P. F. and Diebold, F. X. (1997). Optimal prediction under asymmetric loss. *Econometric theory*, 13(06):808–817.

Chu, L., Gao, H., and Chang, W. (2010). A new feature weighting method based on probability distribution in imbalanced text classification. In *Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on*, volume 5, pages 2335–2339. IEEE.

Chyi, Y. (2003). Classification analysis techniques for skewed class distribution problems. *Master Thesis, Department of Information Management, National Sun Yat-Sen University*.

Cieslak, D. A. and Chawla, N. V. (2008). Learning decision trees for unbalanced data. In *Machine Learning and Knowledge Discovery in Databases*, pages 241–256. Springer.

Cieslak, D. A., Hoens, T. R., Chawla, N. V., and Kegelmeyer, W. P. (2012). Hellinger distance decision trees are robust and skew-insensitive. *Data Mining and Knowledge Discovery*, 24(1):136–158.

Cohen, G., Hilario, M., Sax, H., Hugonnet, S., and Geissbuhler, A. (2006). Learning from imbalanced data in surveillance of nosocomial infection. *Artificial Intelligence in Medicine*, 37(1):7–18.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

Crone, S. F., Lessmann, S., and Stahlbock, R. (2005). Utility based data mining for time series analysis: cost-sensitive learning for neural network predictors. In *Proceedings of the 1st international workshop on Utility-based data mining*, pages 59–68. ACM.

Daskalaki, S., Kopanas, I., and Avouris, N. (2006). Evaluation of classifiers for an uneven class distribution problem. *Applied artificial intelligence*, 20(5):381–417.

Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *ICML'06: Proc. of the 23rd Int. Conf. on Machine Learning*, ACM ICPS, pages 233–240. ACM.

Dehuri, S., Patnaik, S., Ghosh, A., and Mall, R. (2008). Application of elitist multi-objective genetic algorithm for classification rule generation. *Applied Soft Computing*, 8(1):477–487.

Del Castillo, M. D. and Serrano, J. I. (2004). A multistrategy approach for digital text categorization from imbalanced documents. *ACM SIGKDD Explorations Newsletter*, 6(1):70–79.

Denil, M. and Trappenberg, T. (2010). Overlap versus imbalance. In *Advances in Artificial Intelligence*, pages 220–231. Springer.

Derrac, J., Triguero, I., Carmona, C. J., Herrera, F., et al. (2012). Evolutionary-based selection of generalized instances for imbalanced classification. *Knowledge-Based Systems*, 25(1):3–12.

Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D., and Weingessel, A. (2011). *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien.*

Domingos, P. (1999). Metacost: A general method for making classifiers cost-sensitive. In *KDD'99: Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining*, pages 155–164. ACM Press.

Doucette, J. and Heywood, M. I. (2008). Gp classification under imbalanced data sets: Active sub-sampling and auc approximation. In *Genetic Programming*, pages 266–277. Springer.

Drown, D. J., Khoshgoftaar, T. M., and Seliya, N. (2009). Evolutionary sampling and software quality modeling of high-assurance systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 39(5):1097–1107.

Drummond, C. and Holte, R. C. (2000). Exploiting the cost (in) sensitivity of decision tree splitting criteria. In *ICML*, pages 239–246.

Drummond, C., Holte, R. C., et al. (2003). C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on Learning from Imbalanced Datasets II*, volume 11. Citeseer.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *IJCAI'01: Proc. of 17th Int. Joint Conf. of Artificial Intelligence*, volume 1, pages 973–978. Morgan Kaufmann Publishers.

Ertekin, Ş. (2013). Adaptive oversampling for imbalanced data classification. In *Information Sciences and Systems 2013*, pages 261–269. Springer.

Ertekin, S., Huang, J., Bottou, L., and Giles, L. (2007a). Learning on the border: active learning in imbalanced data classification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 127–136. ACM.

Ertekin, S., Huang, J., and Giles, C. L. (2007b). Active learning for class imbalance problem. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 823–824. ACM.

Estabrooks, A. and Japkowicz, N. (2001). A mixture-of-experts framework for learning from imbalanced data sets. In *Advances in Intelligent Data Analysis*, pages 34–43. Springer.

Estabrooks, A., Jo, T., and Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, 20(1):18–36.

Fan, W., Stolfo, S. J., Zhang, J., and Chan, P. K. (1999). Adacost: misclassification cost-sensitive boosting. In *ICML*, pages 97–105. Citeseer.

Fernández, A., del Jesus, M. J., and Herrera, F. (2010). On the 2-tuples based genetic tuning performance for fuzzy rule based classification systems in imbalanced data-sets. *Information Sciences*, 180(8):1268–1291.

Fernández, A., García, S., del Jesus, M. J., and Herrera, F. (2008). A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. *Fuzzy Sets and Systems*, 159(18):2378–2398.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *The Journal of machine learning research*, 3:1289–1305.

Forman, G. and Cohen, I. (2004). Learning from little: Comparison of classifiers given little training. In *Knowledge Discovery in Databases: PKDD 2004*, pages 161–172. Springer.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The annals of statistics*, 19:1–67.

Fumera, G. and Roli, F. (2002). Support vector machines with embedded reject option. In *Pattern Recognition with Support Vector Machines*, pages 68–82. Springer.

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

Galar, M., Fernández, A., Barrenechea, E., and Herrera, F. (2013). Eusboost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling. *Pattern Recognition*.

Gama, J. (2003). Iterative bayes. *Theoretical Computer Science*, 292(2):417–430.

García, S., Cano, J. R., Fernández, A., and Herrera, F. (2006a). A proposal of evolutionary prototype selection for class imbalance problems. In *Intelligent Data Engineering and Automated Learning–IDEAL 2006*, pages 1415–1423. Springer.

García, S., Cano, J. R., and Herrera, F. (2008a). A memetic algorithm for evolutionary prototype selection: A scaling up approach. *Pattern Recognition*, 41(8):2693–2709.

García, S. and Herrera, F. (2009). Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary Computation*, 17(3):275–306.

García, V., Alejo, R., Sánchez, J. S., Sotoca, J. M., and Mollineda, R. A. (2006b). Combined effects of class imbalance and class overlap on instance-based classification. In *Intelligent Data Engineering and Automated Learning–IDEAL 2006*, pages 371–378. Springer.

García, V., Mollineda, R. A., and Sánchez, J. S. (2008b). A new performance evaluation method for two-class imbalanced problems. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 917–925. Springer.

García, V., Mollineda, R. A., and Sánchez, J. S. (2008c). On the k-nn performance in a challenging scenario of imbalance and overlapping. *Pattern Analysis and Applications*, 11(3-4):269–280.

García, V., Mollineda, R. A., and Sánchez, J. S. (2009). Index of balanced accuracy: A performance measure for skewed class distributions. In *Pattern Recognition and Image Analysis*, pages 441–448. Springer.

Garcia, V., Mollineda, R. A., and Sánchez, J. S. (2010). Theoretical analysis of a performance measure for imbalanced data. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 617–620. IEEE.

García, V., Mollineda, R. A., Sánchez, J. S., Alejo, R., and Sotoca, J. M. (2007). When overlapping unexpectedly alters the class imbalance effects. In *Pattern Recognition and Image Analysis*, pages 499–506. Springer.

García, V., Sánchez, J. S., Martín-Félez, R., and Mollineda, R. A. (2012). Surrounding neighborhood-based smote for learning from imbalanced data sets. *Progress in Artificial Intelligence*, 1(4):347–362.

Granger, C. W. (1999). Outline of forecast theory using generalized cost functions. *Spanish Economic Review*, 1(2):161–173.

Guo, H. and Viktor, H. L. (2004a). Boosting with data generation: Improving the classification of hard to learn examples. In *Innovations in Applied Artificial Intelligence*, pages 1082–1091. Springer.

Guo, H. and Viktor, H. L. (2004b). Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *ACM SIGKDD Explorations Newsletter*, 6(1):30–39.

Hall, M. A. (1999). *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato.

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

Hand, D. J. (2009). Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine learning*, 77(1):103–123.

Hart, P. E. (1968). The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, 14:515–516.

He, H., Bai, Y., Garcia, E. A., and Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 1322–1328. IEEE.

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

Hernández-Orallo, J. (2012). Soft (gaussian cde) regression models and loss functions. *arXiv preprint arXiv:1211.1043*.

Hernndez-Orallo, J. (2013). {ROC} curves for regression. *Pattern Recognition*, 46(12):3395 – 3411.

Hido, S., Kashima, H., and Takahashi, Y. (2009). Roughly balanced bagging for imbalanced data. *Statistical Analysis and Data Mining*, 2(5-6):412–426.

Hinton, G. E. (1989). Connectionist learning procedures. *Artificial intelligence*, 40(1):185–234.

Holte, R. C., Acker, L., Porter, B. W., et al. (1989). Concept learning and the problem of small disjuncts. In *IJCAI*, volume 89, pages 813–818. Citeseer.

Hu, J. (2012). Active learning for imbalance problem using l-gem of rbfnn. In *ICMLC*, pages 490–495.

Hu, S., Liang, Y., Ma, L., and He, Y. (2009). Msmote: improving classification performance when training data is imbalanced. In *Computer Science and Engineering, 2009. WCSE'09. Second International Workshop on*, volume 2, pages 13–17. IEEE.

Huang, K., Yang, H., King, I., and Lyu, M. R. (2004). Learning classifiers from imbalanced data based on biased minimax probability machine. In *Computer*

Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, volume 2, pages II–558. IEEE.

Hulse, J. V., Khoshgoftaar, T. M., and Napolitano, A. (2012). A novel noise-resistant boosting algorithm for class-skewed data. In *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, volume 2, pages 551–557. IEEE.

Hwang, J. P., Park, S., and Kim, E. (2011). A new weighted approach to imbalanced data classification problem via support vector machine with quadratic cost function. *Expert Systems with Applications*, 38(7):8580–8585.

Imam, T., Ting, K. M., and Kamruzzaman, J. (2006). z-svm: An svm for improved classification of imbalanced data. In *AI 2006: Advances in Artificial Intelligence*, pages 264–273. Springer.

Japkowicz, N. (2001a). Concept-learning in the presence of between-class and within-class imbalances. In *Advances in Artificial Intelligence*, pages 67–77. Springer.

Japkowicz, N. (2001b). Supervised versus unsupervised binary-learning by feedforward neural networks. *Machine Learning*, 42(1-2):97–122.

Japkowicz, N. (2003). Class imbalances: are we focusing on the right issue. In *Workshop on Learning from Imbalanced Data Sets II*, volume 1723, page 63.

Japkowicz, N. et al. (2000). Learning from imbalanced data sets: a comparison of various strategies. In *AAAI workshop on learning from imbalanced data sets*, volume 68. Menlo Park, CA.

Japkowicz, N., Myers, C., Gluck, M., et al. (1995). A novelty detection approach to classification. In *IJCAI*, pages 518–523.

Japkowicz, N. and Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449.

Jeatrakul, P., Wong, K. W., and Fung, C. C. (2010). Classification of imbalanced data by combining the complementary neural network and smote algorithm. In *Neural Information Processing. Models and Applications*, pages 152–159. Springer.

Jo, T. and Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter*, 6(1):40–49.

Joshi, M. V., Kumar, V., and Agarwal, R. C. (2001). Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, pages 257–264. IEEE.

Kang, P. and Cho, S. (2006). Eus svms: Ensemble of under-sampled svms for data imbalance problems. In *Neural Information Processing*, pages 837–846. Springer.

Karagiannopoulos, M., Anyfantis, D., Kotsiantis, S., and Pintelas, P. (2007). Local cost sensitive learning for handling imbalanced data sets. In *Control & Automation, 2007. MED'07. Mediterranean Conference on*, pages 1–6. IEEE.

Karmaker, A. and Kwek, S. (2006). A boosting approach to remove class label noise. *International Journal of Hybrid Intelligent Systems*, 3(3):169–177.

Kotsiantis, S. and Pintelas, P. (2003). Mixture of expert agents for handling imbalanced data sets. *Annals of Mathematics, Computing & Teleinformatics*, 1(1):46–55.

Kubat, M., Holte, R. C., and Matwin, S. (1998). Machine learning for the detection of oil spills in satellite radar images. *Machine learning*, 30(2-3):195–215.

Kubat, M. and Matwin, S. (1997). Addressing the curse of imbalanced training sets: One-sided selection. In *Proc. of the 14th Int. Conf. on Machine Learning*, pages 179–186. Morgan Kaufmann.

Lanckriet, G. R., Ghaoui, L. E., Bhattacharyya, C., and Jordan, M. I. (2003). A robust minimax approach to classification. *The Journal of Machine Learning Research*, 3:555–582.

Laurikkala, J. (2001). *Improving identification of difficult small classes by balancing class distribution.* Springer.

Lee, H.-j. and Cho, S. (2006). The novelty detection approach for different degrees of class imbalance. In *Neural Information Processing*, pages 21–30. Springer.

Lee, S. S. (1999). Regularization in skewed binary classification. *Computational Statistics*, 14(2):277.

Lee, S. S. (2000). Noisy replication in skewed binary classification. *Computational statistics & data analysis*, 34(2):165–191.

Lee, T.-H. (2008). Loss functions in time series forecasting. *International encyclopedia of the social sciences.*

Li, C. (2007). Classifying imbalanced data using a bagging ensemble variation (bev). In *Proceedings of the 45th annual southeast regional conference*, pages 203–208. ACM.

Li, C., Jing, C., and Xin-tao, G. (2009). An improved p-svm method used to deal with imbalanced data sets. In *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on*, volume 1, pages 118–122. IEEE.

Li, P., Qiao, P.-L., and Liu, Y.-C. (2008). A hybrid re-sampling method for svm learning from imbalanced data sets. In *Fuzzy Systems and Knowledge Discovery, 2008. FSKD'08. Fifth International Conference on*, volume 2, pages 65–69. IEEE.

Liang, G. and Cohn, A. G. (2013). An effective approach for imbalanced classification: Unevenly balanced bagging. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News*, 2(3):18–22.

Ling, C. X., Yang, Q., Wang, J., and Zhang, S. (2004). Decision trees with minimal costs. In *Proceedings of the twenty-first international conference on Machine learning*, page 69. ACM.

Liu, A., Ghosh, J., and Martin, C. E. (2007). Generative oversampling for mining imbalanced datasets. In *DMIN*, pages 66–72.

Liu, T.-Y. (2009). Easyensemble and feature selection for imbalance data sets. In *Bioinformatics, Systems Biology and Intelligent Computing, 2009. IJCBS'09. International Joint Conference on*, pages 517–520. IEEE.

Liu, W., Chawla, S., Cieslak, D. A., and Chawla, N. V. (2010). A robust decision tree algorithm for imbalanced data sets. In *SDM*, volume 10, pages 766–777. SIAM.

Liu, Y., An, A., and Huang, X. (2006). Boosting prediction accuracy on imbalanced datasets with svm ensembles. In *Advances in Knowledge Discovery and Data Mining*, pages 107–118. Springer.

Maciejewski, T. and Stefanowski, J. (2011). Local neighbourhood extension of smote for mining imbalanced data. In *Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium on*, pages 104–111. IEEE.

Maheshwari, S., Agrawal, J., and Sharma, S. (2011). A new approach for classification of highly imbalanced datasets using evolutionary algorithms. *Intl. J. Sci. Eng. Res*, 2:1–5.

Maloof, M. A. (2003). Learning when data sets are imbalanced and when costs are unequal and unknown. In *ICML-2003 workshop on learning from imbalanced data sets II*, volume 2, pages 2–1.

Manevitz, L. and Yousef, M. (2007). One-class document classification via neural networks. *Neurocomputing*, 70(7):1466–1481.

Manevitz, L. M. and Yousef, M. (2002). One-class svms for document classification. *the Journal of machine Learning research*, 2:139–154.

Mani, I. and Zhang, I. (2003). knn approach to unbalanced data distributions: a case study involving information extraction. In *Proceedings of Workshop on Learning from Imbalanced Datasets*.

Martínez-García, J. M., Suárez-Araujo, C. P., and Báez, P. G. (2012). Sneom: a sanger network based extended over-sampling method. application to imbalanced biomedical datasets. In *Neural Information Processing*, pages 584–592. Springer.

McCarthy, K., Zabar, B., and Weiss, G. (2005). Does cost-sensitive learning beat sampling for classifying rare classes? In *Proceedings of the 1st international workshop on Utility-based data mining*, pages 69–77. ACM.

Mease, D., Wyner, A., and Buja, A. (2007). Cost-weighted boosting with jittering and over/under-sampling: Jous-boost. *J. Machine Learning Research*, 8:409–439.

Menardi, G. and Torelli, N. (2010). Training and assessing classification rules with imbalanced data. *Data Mining and Knowledge Discovery*, pages 1–31.

Mi, Y. (2013). Imbalanced classification based on active learning smote. *Research Journal of Applied Sciences*, 5.

Milborrow, S. (2012). *earth: Multivariate Adaptive Regression Spline Models. Derived from mda:mars by Trevor Hastie and Rob Tibshirani.*

Mladenic, D. and Grobelnik, M. (1999). Feature selection for unbalanced class distribution and naive bayes. In *ICML*, volume 99, pages 258–267.

Molinara, M., Ricamato, M. T., and Tortorella, F. (2007). Facing imbalanced classes through aggregation of classifiers. In *Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference on*, pages 43–48. IEEE.

Naganjaneyulu, S. and Kuppa, M. R. (2013). A novel framework for class imbalance learning using intelligent under-sampling. *Progress in Artificial Intelligence*, 2(1):73–84.

Nakamura, M., Kajiwara, Y., Otsuka, A., and Kimura, H. (2013). Lvq-smote–learning vector quantization based synthetic minority over–sampling technique for biomedical data. *BioData mining*, 6(1):16.

Napierała, K., Stefanowski, J., and Wilk, S. (2010). Learning from imbalanced data in presence of noisy and borderline examples. In *Rough Sets and Current Trends in Computing*, pages 158–167. Springer.

Oh, S.-H. (2011). Error back-propagation algorithm for classification of imbalanced data. *Neurocomputing*, 74(6):1058–1061.

Pearson, R., Goney, G., and Shwaber, J. (2003). Imbalanced clustering for microarray time-series. In *Proceedings of the ICML*, volume 3.

Phua, C., Alahakoon, D., and Lee, V. (2004). Minority report in fraud detection: classification of skewed data. *ACM SIGKDD Explorations Newsletter*, 6(1):50–59.

Prati, R. C., Batista, G. E., and Monard, M. C. (2004a). Class imbalances versus class overlapping: an analysis of a learning system behavior. In *MICAI 2004: Advances in Artificial Intelligence*, pages 312–321. Springer.

Prati, R. C., Batista, G. E., and Monard, M. C. (2004b). Learning with class skews and small disjuncts. In *Advances in Artificial Intelligence–SBIA 2004*, pages 296–306. Springer.

Provost, F. J. and Fawcett, T. (1997). Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *KDD*, volume 97, pages 43–48.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Ramentol, E., Caballero, Y., Bello, R., and Herrera, F. (2012a). Smote-rsb*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory. *Knowledge and Information Systems*, 33(2):245–265.

Ramentol, E., Verbiest, N., Bello, R., Caballero, Y., Cornelis, C., and Herrera, F. (2012b). Smote-frst: a new resampling method using fuzzy rough set theory. In *10th International FLINS conference on uncertainty modelling in knowledge engineering and decision making (to appear)*.

Ranawana, R. and Palade, V. (2006). Optimized precision-a new measure for classifier performance evaluation. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 2254–2261. IEEE.

Raskutti, B. and Kowalczyk, A. (2004). Extreme re-balancing for svms: a case study. *ACM Sigkdd Explorations Newsletter*, 6(1):60–69.

Ribeiro, R. P. (2011). *Utility-based Regression*. PhD thesis, Dep. Computer Science, Faculty of Sciences - University of Porto.

Rodriguez, J. J., Diez-Pastor, J. F., Maudes, J., and Garcia-Osorio, C. (2012). Disturbing neighbors ensembles of trees for imbalanced data. In *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, volume 2, pages 83–88. IEEE.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471.

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., and Folleco, A. (2011). An empirical study of the classification performance of learners on imbalanced and noisy software quality data. *Information Sciences*.

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., and Napolitano, A. (2007). Mining data with rare events: a case study. In *Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE International Conference on*, volume 2, pages 132–139. IEEE.

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., and Napolitano, A. (2010). Rusboost: A hybrid approach to alleviating class imbalance. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 40(1):185–197.

Sinha, A. P. and May, J. H. (2004). Evaluating and tuning predictive data mining models using receiver operating characteristic curves. *Journal of Management Information Systems*, 21(3):249–280.

Song, J., Lu, X., and Wu, X. (2009). An improved adaboost algorithm for unbalanced classification data. In *Fuzzy Systems and Knowledge Discovery, 2009. FSKD'09. Sixth International Conference on*, volume 1, pages 109–113. IEEE.

Songwattanasiri, P. and Sinapiromsaran, K. (2010). Smoute: Synthetics minority over-sampling and under-sampling techniques for class imbalanced problem. In *Proceedings of the Annual International Conference on Computer Science Education: Innovation and Technology, Special Track: Knowledge Discovery*, pages 78–83.

Stefanowski, J. and Wilk, S. (2007). Improving rule based classifiers induced by modlem by selective pre-processing of imbalanced data. In *Proc. of the RSKD Workshop at ECML/PKDD, Warsaw*, pages 54–65.

Stefanowski, J. and Wilk, S. (2008). Selective pre-processing of imbalanced data for improving classification performance. In *Data Warehousing and Knowledge Discovery*, pages 283–292. Springer.

Sumadhi, T. and Hemalatha, M. (2013). An enhanced approach for solving class imbalance problem in automatic image annotation. *International Journal of Image, Graphics and Signal Processing (IJIGSP)*, 5(2):9.

Sun, Y., Kamel, M. S., Wong, A. K., and Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378.

Sun, Y., Wong, A. K., and Kamel, M. S. (2009). Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(04):687–719.

Tahir, M. A., Kittler, J., and Yan, F. (2012). Inverse random under sampling for class imbalance problem and its application to multi-label classification. *Pattern Recognition*, 45(10):3738–3750.

Tan, A., Gilbert, D., and Deville, Y. (2003). Multi-class protein fold classification using a new ensemble machine learning approach.

Tang, Y. and Zhang, Y.-Q. (2006). Granular svm with repetitive undersampling for highly imbalanced protein homology prediction. In *Granular Computing, 2006 IEEE International Conference on*, pages 457–460. IEEE.

Tang, Y., Zhang, Y.-Q., Chawla, N. V., and Krasser, S. (2009). Svms modeling for highly imbalanced classification. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(1):281–288.

Tao, D., Tang, X., Li, X., and Wu, X. (2006). Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(7):1088–1099.

Thai-Nghe, N., Gantner, Z., and Schmidt-Thieme, L. (2011). A new evaluation measure for learning from imbalanced data. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 537–542. IEEE.

Ting, K. M. (2000). A comparative study of cost-sensitive boosting algorithms. In *In Proceedings of the 17th International Conference on Machine Learning*. Citeseer.

Tomek, I. (1976). Two modifications of cnn. *IEEE Trans. Syst. Man Cybern.*, (11):769–772.

Torgo, L. (2005). Regression error characteristic surfaces. In *KDD'05: Proc. of the 11th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 697–702. ACM Press.

Torgo, L. (2010). *Data Mining with R, learning with case studies.* CRC Press, Boca Raton, New York, UK.

Torgo, L. and Ribeiro, R. P. (2007). Utility-based regression. In *PKDD'07: Proc. of 11th European Conf. on Principles and Practice of Knowledge Discovery in Databases*, pages 597–604. Springer.

Torgo, L. and Ribeiro, R. P. (2009). Precision and recall in regression. In *DS'09: 12th Int. Conf. on Discovery Science*, pages 332–346. Springer.

Van Der Putten, P. and Van Someren, M. (2004). A bias-variance analysis of a real world learning problem: The coil challenge 2000. *Machine Learning*, 57(1-2):177–195.

Vasu, M. and Ravi, V. (2011). A hybrid under-sampling approach for mining unbalanced datasets: applications to banking and insurance. *International Journal of Data Mining, Modelling and Management*, 3(1):75–105.

Verbiest, N., Ramentol, E., Cornelis, C., and Herrera, F. (2012). Improving smote with fuzzy rough prototype selection to detect noise in imbalanced classification data. In *Advances in Artificial Intelligence–IBERAMIA 2012*, pages 169–178. Springer.

Veropoulos, K., Campbell, C., Cristianini, N., et al. (1999). Controlling the sensitivity of support vector machines. In *Proceedings of the international joint conference on artificial intelligence*, volume 1999, pages 55–60. Citeseer.

Wang, B. X. and Japkowicz, N. (2010). Boosting support vector machines for imbalanced data sets. *Knowledge and information systems*, 25(1):1–20.

Wang, J., Xu, M., Wang, H., and Zhang, J. (2006). Classification of imbalanced data by using the smote algorithm and locally linear embedding. In *Signal Processing, 2006 8th International Conference on*, volume 3. IEEE.

Wang, S. and Yao, X. (2009). Diversity analysis on imbalanced data sets by using ensemble models. In *Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on*, pages 324–331. IEEE.

Wasikowski, M. and Chen, X.-w. (2010). Combating the small sample class imbalance problem using feature selection. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1388–1400.

Weiguo, D., Li, W., Yiyang, W., and Zhong, Q. (2012). An improved svm-km model for imbalanced datasets. In *Industrial Control and Electronics Engineering (ICICEE), 2012 International Conference on*, pages 100–103. IEEE.

Weiss, G. M. (2004). Mining with rarity: a unifying framework. *SIGKDD Explorations Newsletter*, 6(1):7–19.

Weiss, G. M. (2005). Mining with rare cases. In *Data Mining and Knowledge Discovery Handbook*, pages 765–776. Springer.

Weiss, G. M. (2010). The impact of small disjuncts on classifier learning. In *Data Mining*, pages 193–226. Springer.

Weiss, G. M. and Hirsh, H. (2000). A quantitative study of small disjuncts. In *AAAI/IAAI*, pages 665–670.

Weiss, G. M. and Provost, F. J. (2003). Learning when training data are costly: the effect of class distribution on tree induction. *J. Artif. Intell. Res.(JAIR)*, 19:315–354.

Whitley, L. D., Beveridge, J. R., Guerra-Salcedo, C., and Graves, C. R. (1997). Messy genetic algorithms for subset feature selection. In *ICGA*, pages 568–575.

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

Wu, G. and Chang, E. Y. (2003). Class-boundary alignment for imbalanced dataset learning. In *ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*, pages 49–56.

Wu, G. and Chang, E. Y. (2005). Kba: Kernel boundary alignment considering imbalanced data distribution. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):786–795.

Xiao, J., Xie, L., He, C., and Jiang, X. (2012). Dynamic classifier ensemble model for customer classification with imbalanced class distribution. *Expert Systems with Applications*, 39(3):3668–3675.

Xuan, L., Zhigang, C., and Fan, Y. (2013). Exploring of clustering algorithm on class-imbalanced data. In *Computer Science & Education (ICCSE), 2013 8th International Conference on*, pages 89–93. IEEE.

Yan, R., Liu, Y., Jin, R., and Hauptmann, A. (2003). On predicting rare classes with svm ensembles in scene classification. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 3, pages III–21. IEEE.

Yang, Z. and Gao, D. (2012). An active under-sampling approach for imbalanced data classification. In *Computational Intelligence and Design (ISCID), 2012 Fifth International Symposium on*, volume 2, pages 270–273. IEEE.

Yen, S.-J. and Lee, Y.-S. (2006). Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset. In *Intelligent Control and Automation*, pages 731–740. Springer.

Yen, S.-J. and Lee, Y.-S. (2009). Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications*, 36(3):5718–5727.

Yong, Y. (2012). The research of imbalanced data set of sample sampling method based on k-means cluster and genetic algorithm. *Energy Procedia*, 17:164–170.

Yoon, K. and Kwek, S. (2005). An unsupervised learning approach to resolving the data imbalanced issue in supervised learning problems in functional genomics. In *Hybrid Intelligent Systems, 2005. HIS'05. Fifth International Conference on*, pages 6–pp. IEEE.

Yuanhong, D., Hongchang, C., and Tao, P. (2009). Cost-sensitive support vector machine based on weighted attribute. In *Information Technology and Applications, 2009. IFITA'09. International Forum on*, volume 1, pages 690–692. IEEE.

Zadrozny, B. (2005). One-benefit learning: cost-sensitive learning with restricted cost information. In *UBDM'05: Proc. of the 1st Int. Workshop on Utility-Based Data Mining*, pages 53–58. ACM Press.

Zadrozny, B. and Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 694–699. ACM.

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

Zellner, A. (1986). Bayesian estimation and prediction using asymmetric loss functions. *Journal of the American Statistical Association*, 81(394):446–451.

Zhang, C.-X., Wang, G.-W., Zhang, J.-S., and Guo, G. (2013). Irusrt: A novel imbalanced learning technique by combining inverse random under sampling and random tree. *Communications in Statistics-Simulation and Computation*, (just-accepted).

Zhang, D., Liu, W., Gong, X., and Jin, H. (2011). A novel improved smote resampling algorithm based on fractal. *Journal of Computational Information Systems*, 7(6):2204–2211.

Zhao, H., Sinha, A. P., and Bansal, G. (2011). An extended tuning method for cost-sensitive regression and forecasting. *Decision Support Systems*, 51(3):372–383.

Zheng, Z., Wu, X., and Srihari, R. (2004). Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter*, 6(1):80–89.

Zhou, Z.-H. and Liu, X.-Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *Knowledge and Data Engineering, IEEE Transactions on*, 18(1):63–77.

Zhu, J. and Hovy, E. H. (2007). Active learning for word sense disambiguation with methods for addressing the class imbalance problem. In *EMNLP-CoNLL*, volume 7, pages 783–790.

Zhuang, L. and Dai, H. (2006). Parameter estimation of one-class svm on imbalance text classification. In *Advances in Artificial Intelligence*, pages 538–549. Springer.