ENSEMBLE DIVERSITY FOR CLASS IMBALANCE LEARNING

by

SHUO WANG

A thesis submitted to
The University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

School of Computer Science
College of Engineering and Physical Sciences
The University of Birmingham
July 2011

Abstract

This thesis studies the diversity issue of classification ensembles for class imbalance learning problems. Class imbalance learning refers to learning from imbalanced data sets, in which some classes of examples (minority) are highly under-represented comparing to other classes (majority). The very skewed class distribution degrades the learning ability of many traditional machine learning methods, especially in the recognition of examples from the minority classes, which are often deemed to be more important and interesting. Although quite a few ensemble learning approaches have been proposed to handle the problem, no in-depth research exists to explain why and when they can be helpful. Our objectives are to understand how ensemble diversity affects the classification performance for a class imbalance problem according to single-class and overall performance measures, and to make best use of diversity to improve the performance.

As the first stage, we study the relationship between ensemble diversity and generalization performance for class imbalance problems. We investigate mathematical links between single-class performance and ensemble diversity. It is found that how the single-class measures change along with diversity falls into six different situations. These findings are then verified in class imbalance scenarios through empirical studies. The impact of diversity on overall performance is also investigated empirically. Strong correlations between diversity and the performance measures are found. Diversity shows a positive impact on the recognition of the minority class and benefits the overall performance of ensembles in class imbalance learning. Our results help to understand if and why ensemble diversity can help to deal with class imbalance problems.

Encouraged by the positive role of diversity in class imbalance learning, we then focus on a specific ensemble learning technique, the negative correlation learning (NCL) algorithm, which considers diversity explicitly when creating ensembles and has achieved great empirical success. We propose a new learning algorithm based on the idea of NCL, named AdaBoost.NC, for classification problems. An "ambiguity" term decomposed from the 0-1 error function is introduced into the training framework of AdaBoost. It demonstrates superiority in both effectiveness and efficiency. Its good generalization performance is explained by theoretical and empirical evidences. It can be viewed as the first NCL algorithm specializing in classification problems.

Most existing ensemble methods for class imbalance problems suffer from the problems of overfitting and over-generalization. To improve this situation, we address the class imbalance issue by making use of ensemble diversity. We investigate the generalization ability of NCL algorithms, including AdaBoost.NC, to tackle two-class imbalance problems. We find that NCL methods integrated with random oversampling are effective in recognizing minority class examples without losing the overall performance, especially the AdaBoost.NC tree ensemble. This is achieved by providing smoother and less overfitting classification boundaries for the minority class. The results here show the usefulness of diversity and open up a novel way to deal with class imbalance problems.

Since the two-class imbalance is not the only scenario in real-world applications, multi-class imbalance problems deserve equal attention. To understand what problems multi-class can cause and how it affects the classification performance, we study the multi-class difficulty by analyzing the multi-minority and multi-majority cases respectively. Both lead to a significant performance reduction. The multi-majority case appears to be more harmful. The results reveal possible issues that a class imbalance learning technique could have when dealing with multi-class tasks. Following this part of analysis and the promising results of AdaBoost.NC on two-class imbalance problems, we apply AdaBoost.NC to a set of multi-class imbalance domains with the aim of solving them effectively and directly. Our method shows good generalization in minority classes and balances the performance across different classes well without using any class decomposition schemes.

Finally, we conclude this thesis with how the study has contributed to class imbalance learning and ensemble learning, and propose several possible directions for future research that may improve and extend this work.

ACKNOWLEDGEMENTS

Foremost, I would like to express my deep and sincere gratitude to my supervisor Prof. Xin Yao for his inspiration, patience and encouragement. He introduced me to the field of machine learning and provided lots of good ideas and advice. I would have never completed this thesis without his help. He educated me generously in how to be a good researcher with his immense knowledge. The joy and enthusiasm he has for his research were contagious and motivational for me to hurdle all the obstacles.

Besides Xin, I am deeply grateful to my research committee members, Dr. Peter Tino and Dr. Ata Kaban, for their insightful comments and hard questions throughout my Ph.D study. Their great ideas and unfailing support guided me to go further in this field. Their wide knowledge and logical way of thinking have been of great value for me.

My sincere thanks also go to Dr. Steven Vickers, who took effort in my Ph.D progress and helped with various applications, and Prof. Achim Jung, Prof. John Barnden and Dr. Nick Hawes, who offered me great opportunity to be a good demonstrator. It was an enjoyable experience of teaching.

I wish to thank the examiners of this thesis, Dr. Wenjia Wang and Dr. Ata Kaban, for agreeing to be the examiners of my Ph.D viva.

Luckily for me, I have benefited from numerous research fellows who turned into even better friends: Huanhuan Chen, Shan He, Leandro Minku, Pietro Oliveto, Ke Tang, Andrea Soltoggio and Arjun Chandra. They gave me wise advice and helped me along the way.

I am indebted to my many Ph.D colleagues for providing a stimulating and fun environment in which to learn and grow. I am especially grateful to Seyyed Shah, Rodrigo

Soares, Guanzhou Lu, Fubiao Xia and Haobo Fu for their support, company and laughs.

I would like to thank my best friends Siripen Songprasit and Jiayi Lee, for helping me get through the most difficult times, and for all the emotional support, entertainment, and caring they provided.

Last but not the least, I would like to thank my family: my parents Subin Wang and Ruiping Dang, for all their love and encouragement. They supported me in all my pursuits and put me where I am today. To them I dedicate this thesis.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF NOTATIONS

X	Instance space
Y	A finite label set with c labels $\{\omega_1, \ldots, \omega_c\}$
x_j	Any input example
y_j	The true label of x_j
Z	A data set composed of (x_j, y_j) pairs
$\lvert Z \rvert$	The cardinality of data set Z
Z_{min}	The subset of Z composed of (x_j, y_j) pairs where y_j is the minority class label
Z_{maj}	The subset of Z composed of (x_j, y_j) pairs where y_j is the majority class label
S	The obtained data set after Z is resampled
S_{min}	The subset of S composed of (x_j, y_j) pairs where y_j is the minority class label
S_{maj}	The subset of S composed of (x_j, y_j) pairs where y_j is the majority class label
f_i	A single classifier
$\bar{f}$	An ensemble composed of L individual classifiers $\{f_1, f_2, \ldots, f_L\}$
$O_{j,i}$	The oracle output of f_i on example x_j
$O_{j,ens}$	The oracle output of the ensemble on example x_j
θ	The imbalance rate of the training data
Q	Q-statistic
GD	Generalized diversity
R	Single-class measure recall
P	Single-class measure precision
F	Single-class measure F-measure
P_{ovr}	Overall accuracy of the ensemble

p .... Overall accuracy of the individual classifier

p_{ab} ...... The occurrence probability of the respective combination of correct and wrong predictions for any pair of classifiers

amb .... The ambiguity term

λ .... The penalty strength in NCL algorithms

CHAPTER 1

INTRODUCTION

In machine learning and pattern recognition, classification is an important and popular research area with the task of assigning a given data input into one of finite categories (i.e. a set of *class labels*), such as speech recognition, email spam prediction and defect prediction in language programming. Given a training set of examples with definite labels, a range of supervised learning algorithms have been well developed for learning from the data. The resulting model (i.e. *a classifier*) will enable us to predict the outcome for new unseen examples. For instance, decision trees, neural networks and support vector machines have been successfully applied to many real-world classification problems. The performance of a classifier is judged by its *generalization* ability of predicting unseen data. The overall accuracy or error rate is often measured as the evaluation criterion.

Many of current learning algorithms implicitly assume that classification errors coming from different classes have the same cost, and treat them equally during learning. Their learning procedure is driven by maximizing the overall accuracy (Visa and Ralescu, 2005; Gu et al., 2008; Guo et al., 2008; Provost, 2000; He and Garcia, 2009). However, this is not always the case in real-world applications. A number of problems have unequal misclassification costs among classes, such as fraud detection in user behaviour and business transactions (Fawcett and Provost, 1997; Chan and Stolfo, 1998), risk management in telecommunications (Ezawa et al., 1996), medical diagnosis (Valdovinos and Sanchez, 2005) and defect prediction in software engineering (Menzies et al., 2007). In such problems, some classes of data are heavily under-represented compared to other classes. For the example of defect prediction for software quality, a defect case is much less likely to occur than non-defect ones. The rare examples in specific classes are often more costly and important. The failure of predicting these classes can cause incalculable loss. This situation is known as the *class imbalance problem*. The standard classifiers with the aim of maximum overall accuracy often have poor performance on this problem, because small classes contribute little (Visa and Ralescu, 2005). They ignore different types of misclassification errors (Gu et al., 2008).

For example, a classifier learns from a data set consisting of a class with 98 examples and a second class with only 2 examples. If all data are labelled as the first class, the classifier still achieves a 98% accuracy (Visa and Ralescu, 2005). Apparently, it is useless in practice.

The difficulty of handling class imbalance problems has been drawing growing interest from academia and industry. Numerous solutions were proposed at the data and algorithm levels. Data-level methods include different forms of resampling techniques, which rebalance skewed class distributions by manipulating training data directly, such as random oversampling, random undersampling, heuristic sampling techniques and their combinations. They are easy to use, but overfitting and over-generalization problems have been reported. At the algorithm level, solutions adapt algorithms to enforce the learning with regard to the small classes. The key idea is to adjust the inductive bias of a classifier internally (Guo et al., 2008; Zadrozny and Elkan, 2001; Wu and Chang, 2003; Lin et al., 2002). The weak point is that the treatment is algorithm-specific. Cost-sensitive learning methods have been applied to class imbalance applications when the misclassification costs are known (Gu et al., 2008; Zhou and Liu, 2006b; Monard and Batista, 2002). They assume that rare examples have higher costs, and the cost information is introduced into the learning procedure with the aim of minimizing the total cost. In many cases, however, explicit costs are not available.

Since ensemble learning showed great success in machine learning, it has become another major technique of handling class imbalance problems. The idea of ensemble learning is to build a set of learners on the same task and then combine them to form a better one (Hansen and Salamon, 1990). Particularly, Bagging (Breiman, 1996) and Boosting (Freund and Schapire, 1996) are the most popular ensemble learning methods in the literature (Opitz and Maclin, 1999). They have been modified and widely used in class imbalance learning for their flexibility and effectiveness, such as BEV (Li, 2007), SMOTEBoost (Chawla et al., 2003), RareBoost (Joshi et al., 2001), etc. From the individual level, an ensemble can be integrated with other class imbalance learning techniques easily. From the ensemble level, combining multiple classifiers can stabilize the prediction and achieve better generalization.

Through the combination of multiple learners, maintaining a certain level of diversity in their predictions is important for the success of ensembles, which has been explained from both theoretical (Krogh and Vedelsby, 1995; Brown et al., 2005) and empirical (Liu and Yao, 1999a; Islam et al., 2003) aspects. It is evident that no gain can be achieved from combining a set of identical models. However, it is still not clear about the role of diversity in the benefit of using ensembles to deal with class imbalance problems. In addition, most existing ensemble methods in class imbalance learning are restricted to the use of data-level techniques and cost-sensitive strategies, which suffer from some known drawbacks. Therefore, an in-depth study of the effect of ensembles in the context of class imbalance learning is necessary. It can help us to understand their learning potential and allow us to make best use of this effect for better solutions.

Considering the great efforts of ensemble learning methods in class imbalance learning and the importance of diversity to an ensemble model, this thesis gives a thorough study of ensemble diversity for class imbalance learning and inspires a new ensemble algorithm for classification that promotes diversity explicitly. It is used to solve imbalanced problems with certain advantages and success.

In this chapter, sections 1.1 and 1.2 state the problems addressed in this thesis, including a brief idea about where the problems come from. Section 1.3 gives a clear explanation of the research questions, their motivations and why they are important. It is followed by an overview of the contributions in section 1.4. Section 1.5 presents the thesis structure along with a chapter-by-chapter summary.

1.1 Class Imbalance Learning

Class imbalance learning refers to learning from data sets that exhibit significant imbalances among or within classes. Any data set with uneven data distributions can be considered imbalanced. The common understanding about "imbalance" in the literature is concerned with the *between-class imbalance*, in which case some classes of data are highly under-represented compared to other classes (He and Garcia, 2009). By convention, we call the classes having more examples the *majority* classes, and the ones having fewer examples the *minority* classes. Misclassifying an example from the minority class is usually more costly. For example, in a defect prediction problem from software engineering, codes with defects are much less likely to occur than codes without defects. The defect class is thus the minority and causes more concern in how to recognize defects accurately. In practical applications, the ratio between classes can be drastic as 100:1, 1000:1 and 10000:1 (Pearson et al., 2003; Wu and Chang, 2003). It is worth mentioning that although a lot of efforts have been devoted to two-class imbalance problems, multi-class imbalance is of equal importance, which involves more than one minority or majority class. It exists in many real-world domains, such as protein fold classification (Tan et al., 2003; Zhao et al., 2008) and weld flaw classification (Liao, 2008).

The imbalance can also happen within a class, referred to as the *within-class imbalance*. In this case, a class is composed of a number of sub-clusters and some sub-clusters have much fewer examples than other sub-clusters (Japkowicz, 2001a). Although under-represented sub-clusters can occur to both minority and majority classes, they are more likely to exist in the minority class, since it is often much easier to collect examples for the majority class (Weiss, 2004). For a clear understanding, we clarify that the "imbalance" without an explicit description in the rest of the thesis indicates the between-class imbalance.

The imbalanced distribution pervasively exists in real-world applications. Although it is often and should be the focus of research efforts, some studies have shown that the minority concept can be learnt quite well for certain imbalanced data sets with simple class distributions, such as a linearly separable problem (Japkowicz and Stephen, 2002; Japkowicz, 2003; Prati et al., 2004; Batista et al., 2004, 2005). It suggests that the imbalance degree is not the only factor responsible for the performance reduction of learning algorithms. A learner's sensitivity to the class imbalance was found to depend on the data complexity and the overall size of the training set.

Data complexity comprises issues such as overlapping (Batista et al., 2005; Prati et al., 2004) and small disjuncts (Holte et al., 1989; Jo and Japkowicz, 2004). The degree of overlapping between classes and how the minority class examples distribute in data space aggravate the negative effect of class imbalance. The small disjuncts problem is also associated with the within-class imbalance (Japkowicz, 2003; Japkowicz and Stephen, 2002). In terms of the size of the training data, a very large domain has a good chance that the minority class is represented by a reasonable number of examples, and thus may be less affected by imbalance than a small domain containing very few minority class examples. In other words, the rarity of the minority class can be in a relative or absolute sense in terms of the number of available examples (He and Garcia, 2009).

Even if the degree of imbalance may not be the direct reason for the performance reduction in the above situations, they have the same radical issue caused by the imbalanced distribution: the relatively or absolutely underrepresented class cannot draw equal attention to the learning algorithm compared to the majority class, which results in very specific classification rules for the minority class without much generalization ability for future prediction or missing rules for the small concepts (Weiss, 2004; He and Garcia, 2009). This is because the learner assumes or expects balanced data distributions and equal misclassification costs (Visa and Ralescu, 2005; Gu et al., 2008; Guo et al., 2008;

Provost, 2000; He and Garcia, 2009). How to better recognize examples from the minority class causes the major concern in class imbalance learning. It has been studied by either rebalancing the data distribution or adjusting the learning bias inside the algorithm. Considering the learning difficulty of an imbalanced problem and the high misclassification cost of minority class examples, the learning objective of class imbalance learning can be generally described as "obtaining a classifier that will provide high accuracy for the minority class without severely jeopardizing the accuracy of the majority class" (He and Garcia, 2009).

Another related study is cost-sensitive learning. Given misclassification costs for each class, a cost-sensitive method takes the cost information into consideration and treats misclassifications differently during the training procedure. It is natural to apply cost-sensitive learning methods to class imbalance problems by assigning a greater cost to the minority class. Maloof showed that learning from imbalanced data sets and learning with unequal costs can be handled in a similar manner (Maloof, 2003). A strong connection has been indicated between cost-sensitive learning and class imbalance learning in recent research (Liu and Zhou, 2006; Zhou and Liu, 2006b; Monard and Batista, 2002). The cost matrix plays the essential role in cost-sensitive methods, a numerical representation of the penalty when an example is mislabelled from one class to another (Elkan, 2001). In many real-world problems, unfortunately, such cost information is not available.

This thesis aims to resolve the fundamental issue of class imbalance learning and develop an effective solution, which has good generalization ability to discriminate the minority class examples from the majority class examples and is applicable to most class imbalance problems.

1.2 Ensemble Learning

The idea of ensemble learning is to employ multiple learners on a given problem and combine their outputs as a "*committee*" to make a final decision for better accuracy. The individual committee member is sometimes called *base learner*. In classification, ensemble learning is also referred to as multiple classifier system (Ho et al., 1994), classifier fusion (Kuncheva, 2002), committee of classifiers, classifier combination, etc. The members' prediction might be real-valued numbers, class labels, posterior probabilities, or any other quality. To make best use of the strengths of the individuals and make up their weaknesses, how to combine the predictions is important and has been studied in the literature (Ho et al., 1994; Kuncheva, 2002; Shipp and Kuncheva, 2002a; Kittler et al., 1998), including averaging, majority vote and probabilistic methods (Kuncheva, 2004; Polikar, 2006; Rao, 2001).

Ensemble learning methods have some desirable features, which encourage the rapid growth of related research. For theoretical reasons (Dietterich, 2000b; Polikar, 2006), every single learning model has limitations and may perform differently due to insufficient data. Averaging many of them can reduce the overall risk of making a poor prediction (Perrone and Cooper, 1993). It is an innate behaviour to consult opinions from others before a decision is made. Second, certain learning algorithms confront the local optima problem, such as neural networks and decision trees. Ensembles may avoid it by running local search from different views, where a better approximation to the true function is expected. Besides, some problems are just too difficult and complex which are beyond the learning ability of the chosen models. Ensembles allow partition of the data space, where each individual only learns from one of the smaller and simpler sub-problems. Their combination can then represent the whole problem better.

For practical reasons (Polikar, 2006), real-world problems can be very large or small. A large data set can be divided into several smaller subsets, which will be processed by multiple learners in parallel. In the case of too little data, resampling techniques can be used in ensembles for drawing overlapping subsets to emphasize the available data. Ensemble methods have also been applied to data fusion applications, where the obtained data come from different sources with heterogeneous features. In protein classification, for example, protein sequences are often extracted into different feature spaces. Each feature space can be used to create a classifier, many of which with different views are then combined as the complement to each other (Zhao et al., 2008).

With the above advantages, ensemble learning has made great contribution to various fields, such as face recognition (Huang et al., 2000), speech recognition (Kirkland, 1995), image analysis (Cherkauer, 1996) and handwritten digit recognition (Hansen et al., 1992).

A lot of efforts have been put into learning algorithms for constructing an effective ensemble model since the 1990s. Particularly, Bagging (Breiman, 1996) and AdaBoost (Freund and Schapire, 1996) have received the most attention in the literature, based on which numerous variations have appeared for different learning scenarios. There also exist other popular methods, such as Random Subspace (Ho, 1998), Random Forests (Breiman, 2001) and Negative Correlation Learning (abbr. NCL) (Liu and Yao, 1999a).

The principle of designing a good ensemble learning algorithm is to consider "*diversity*" among the committee members with good individual accuracies maintained. There is no point to combine a set of identical learners. The concept of "diversity" is generally described as the degree of disagreement or complementarity within an ensemble (Kuncheva, 2003; Kuncheva and Whitaker, 2003; Brown et al., 2005). All ensemble algorithms attempt to encourage diversity. It can be achieved implicitly by manipulating training data or using different training parameters for each individual (Polikar, 2006; Dietterich, 2002), such as Bagging and Random Subspace. It can be enforced explicitly by using NCL algorithms.

Due to different types of error functions and combining rules, ensemble diversity has been studied for regression and classification tasks separately. While it has been clearly formulated in regression ensembles (Brown, 2004; Brown et al., 2005), diversity in classification ensembles is still not fully understood theoretically (Brown, 2010; Kuncheva, 2003). Because class imbalance learning is concerned with a type of classification problems, the thesis will provide further investigations into diversity for classification ensembles and an in-depth analysis of how it can benefit class imbalance learning.

1.3 Research Questions

The focus of this thesis can be simply described as “exploring and exploiting diversity of ensemble learning methods for class imbalance learning”. This section explains the research questions answered by this work around the central point, the motivations behind them, and the reasons why they are important and interesting.

1.3.1 Ensemble Diversity for Class Imbalance Learning

In recent years, ensemble learning methods have become a popular way of advancing the classification of imbalanced data. They can be easily adapted for emphasizing the minority class by using resampling techniques from the data level (Li, 2007; Liu et al., 2009; Chawla et al., 2003) or by applying different misclassification costs from the algorithm level (Joshi et al., 2001; Chawla and Sylvester, 2007; Guo and Viktor, 2004; Fan et al., 1999). In addition, the idea of combining multiple classifiers itself can lower down the risk of overfitting and reduce the error variance (Brown et al., 2005). Particular techniques, such as oversampling and undersampling, are often used with ensembles to improve the generalization of predicting the minority class. For example, SMOTE-Boost (Chawla et al., 2003) is a frequently-used ensemble method that generates new training data for the minority class to adjust the learning bias towards the majority class, and utilizes Boosting (Schapire, 1999) to maintain the accuracy over the entire data set.

Although existing ensemble methods have shown great learning advantages and empirical success over other types of solutions, no explanations have been offered as to why it performs well in class imbalance problems. Some works mention that it is attributed to the difference of individual classifiers (Estabrooks et al., 2004; Kotsiantis and Pintelas, 2003; Chawla and Sylvester, 2007), which is concerned with ensemble diversity. Since diversity has been proved to be one of the main reasons for the success of an ensemble, it is natural to ask if and how class imbalance learning benefits from ensemble methods in terms of diversity. No study has actually investigated diversity in depth regarding its definitions and effects in the context of class imbalance learning. It is unclear whether diversity will have a similar or different impact on the performance of minority/majority classes.

In addition, all existing studies of ensemble diversity are limited to its relationship to overall accuracy or error rate. How it relates to other evaluation criteria has not caused much attention. In imbalanced scenarios, apparently, only considering overall accuracy is not sufficient and less meaningful (Weiss, 2004; Maloof, 2003; Provost and Fawcett, 1997; Joshi, 2002). It is highly sensitive to changes in data. Other performance measures are adopted in the class imbalance learning literature. To assess the performance over a specific class we are concerned with, some single-class performance are defined, including recall, precision and F-measure (Rijsbergen, 1979). For overall performance evaluation, ROC/AUC (Fawcett, 2003, 2006; Ling et al., 2003) and G-mean (Kubat et al., 1997) are the most frequently used metrics. Therefore, it would be useful to establish the relationship between diversity and these performance measures.

For the above reasons, the thesis firstly studies the questions of "*what is the relationship between ensemble diversity and the performance measures used in class imbalance learning? Is introducing diversity beneficial to the classification of the minority /majority classes in the presence of imbalanced data?*".

From the viewpoint of gaining a better understanding of the role of ensemble diversity in class imbalance learning, if diversity can be shown to be capable of alleviating the classification difficulty of imbalanced problems, then a new possible way of handling this problem could be to consider diversity explicitly during learning. From the viewpoint of performance evaluation, it is always useful to understand the behaviours of major performance measures caused by diversity, which can provide guidance for designing and choosing a good ensemble model.

To answer the questions, we study the theoretical link between diversity and single-class performance based on several classification patterns in chapter 3. In the same chapter, we also examine how diversity affects the single-class and overall performance through extensive experimental analyses on artificial and real-world imbalanced data sets with highly skewed class distributions. A positive effect of diversity is found in solving class imbalance problems.

1.3.2 Ensemble Methods for Class Imbalance Learning

A variety of ensemble methods have been proposed to deal with class imbalance problems based on Bagging and Boosting, and show superiority in certain applications. Although their standard forms are not very effective in imbalanced scenarios, they can be easily modified to work with other techniques, such as oversampling and undersampling, in order to emphasize small classes. The common strategy for Bagging-based methods is to undersample data from the majority class, and then build component classifiers with relatively balanced training subsets. AdaBoost, the most well-known member in Boosting family (Schapire, 2002), is often modified by employing the advanced oversampling strategy at each step of sequential training. It has also been made cost-sensitive by manipulating its weight-updating rule, which assigns higher costs to rare examples than common ones (Sun et al., 2007).

However, both resampling-based and cost-sensitive ensemble solutions suffer from some known drawbacks. Undersampling majority class examples could abandon potentially useful information (He and Garcia, 2009). Besides, it is always an issue of deciding how many and which examples should be removed. Advanced oversampling techniques often require careful parameter settings to manipulate training data before use. For example, some data generation methods were reported to suffer from the over-generalization problem (He and Garcia, 2009), depending on the setting for creating synthetic examples. The major reason for these problems is that they address "imbalance" by changing training data directly, which could be risky sometimes. Cost-sensitive methods do not work on the data level, but demand clear cost information of classes prior to learning, which is not available in most real-world applications. Generally speaking, it is always a crucial and problem-dependent issue of choosing the best sampling technique and parameter settings for these methods.

Hence, we are motivated to seek an alternative way to overcome the problems of existing ensemble methods. Meanwhile, it should have good generalization especially in the minority class. Encouraged by the results in chapter 3 where diversity shows a positive impact on the minority-class and overall performance, we ask *if and how we can take advantage of ensemble diversity to better deal with class imbalance problems.* Following this line, we devote special attention to negative correlation learning (NCL), a successful ensemble technique that encourages diversity explicitly during training and presents very good generalization ability with a solid theoretical grounding (Liu and Yao, 1999a,b; Brown et al., 2005). A further related question is "*Can NCL methods be good solutions to class imbalance problems?*". At least, they have the advantages of improving generalization without changing training data and requiring much prior data knowledge.

There has been very little work on this topic yet, which will be answered in this thesis. On one hand, it gives a further understanding of the effect of ensemble diversity in solving a specific and important type of classification problems. On the other hand, it provides a new way of dealing with real-world problems that suffer from the class imbalance difficulty. It opens up a practical and novel use of ensemble learning algorithms.

In chapter 5, we explore and exploit diversity through NCL methods to facilitate class imbalance learning with comprehensive and systematic analyses. In particular, we study the effectiveness of AdaBoost.NC, a new NCL method for classification ensembles, in the presence of imbalanced data in depth. More background and motivations about this algorithm can be found in the next section. The algorithm is proposed in chapter 4.

Another emerging research area in class imbalance learning is concerned with multi-class imbalance problems, where multiple minority or/and majority classes exist in data. Most current efforts are focused on two-class tasks. In practice, many applications have more than two classes with uneven class distributions, such as protein fold classification (Zhao et al., 2008; Chen et al., 2006; Tan et al., 2003) and weld flaw classification (Liao, 2008). These multi-class imbalance problems pose new challenges that are not observed in two-class problems and have not been addressed so far. Many useful techniques for two-class tasks were found ineffective on multi-class tasks (Zhou and Liu, 2006b). More investigations are necessary to explain *what problems multi-class can cause and how it affects the classification performance.* Among limited solutions for multi-class imbalance problems, most attention in the literature has been devoted to class decomposition, which breaks the whole problem into several binary sub-problems. It simplifies the problem. However, each individual classifier is trained without full data knowledge. It can cause classification ambiguity or uncovered data regions with respect to each type of decomposition (Tan et al., 2003; Jin and Zhang, 2007; Valizadegan et al., 2008). It is desirable to develop a more effective method without increasing learning problems. According to the current progress in this topic, we would like to *explore new approaches to tackling the multi-class difficulties in class imbalance learning effectively and directly.*

In chapter 6, we study the impact of two types of multi-class, i.e. multi-minority and multi-majority, on the performance of two basic resampling strategies that are widely used in two-class imbalance problems. Based on the results, we propose to use the best NCL strategy obtained in chapter 5 to handle multi-class imbalance problems.

1.3.3 Negative Correlation Learning for Classification Ensembles

Motivated by the positive role of ensemble diversity in class imbalance learning found in chapter 3 of this thesis, we study negative correlation learning (NCL) (Liu and Yao, 1997), an important ensemble learning technique that considers ensemble diversity explicitly during the learning procedure, with the aim of improving the generalization on the minority class. The fundamental part of NCL is to introduce a diversity-related term derived from the generalization error into the ensemble training to manage the accuracy-diversity trade-off. Because of different types of error functions and combining methods, there is no agreed definition and theoretical framework of diversity for classification ensembles. Hence, it is hard to explain and extend NCL to classification problems theoretically.

From the viewpoint of practical use, existing NCL algorithms suffer from some known drawbacks that hinder them from being more widely used. The original NCL algorithm is called Cooperative ensemble learning system (CELS) (Liu and Yao, 1999b), following which several variations have appeared (Liu et al., 2000; Chan and Kasabov, 2005a; Islam et al., 2003; Alam and Islam, 2007; Chen and Yao, 2009). They are all designed for and tested on neural networks. Other base learners have not been applied successfully so far. Besides, they can be computationally expensive for large data sets and large ensembles due to the simultaneous and back-propagation training procedure (Chan and Kasabov, 2005a). In short, low flexibility and poor efficiency are their major problems.

To introduce the idea of NCL into class imbalance learning, we need a better NCL algorithm for classification ensembles, which is flexible, efficient, and as effective as other NCL methods in generalization. With this goal in mind, two questions have to be answered: *1) how to express diversity of a classification ensemble? 2) How to introduce the diversity term in the training procedure without losing flexibility and efficiency?*

In chapter 4, we propose a new NCL algorithm in the classification context, called AdaBoost.NC, in which an "ambiguity" term is obtained from the 0-1 error function to express the difference within the ensemble and introduced into the training framework of AdaBoost. AdaBoost.NC combines the strength of NCL and AdaBoost, and shows very promising results in both effectiveness and efficiency. We provide theoretical and empirical evidences to explain its generalization performance. With these encouraging results, it is explored and exploited as a potential solution for class imbalance problems in chapters 5 and 6.

To our best knowledge, it is the first work that uses the idea of NCL in the design of classification ensembles. It is applicable to all classification tasks and free of choosing any base learner.

1.4 Thesis Contributions

How this thesis contributes to the fields of ensemble learning and pattern classification is summarized here. It will be discussed in more detail in the Conclusions chapter.

- The first systematic and in-depth study of the relationship between diversity of classification ensembles and single-class performance, including the possible behaviours of three single-class performance measures widely used in class imbalance learning as diversity varies. (chapter 3)

- Empirical work demonstrating the correlation between diversity and generalization in terms of both single-class and overall performance in class imbalance scenarios, where strong correlations are found. Diversity exhibits a positive impact on the minority class. It is also beneficial to the overall performance of class imbalance problems. (chapter 3)

- A new learning algorithm for classification ensembles, which is shown to be flexible, effective and efficient based on thorough theoretical explanations and empirical discussions. (chapter 4)

- A novel way of dealing with class imbalance problems by exploiting the new ensemble algorithm, which shows good ability to recognize minority class examples and balance the performance among classes. A comprehensive and in-depth analysis is conducted over artificial and real-world imbalanced data problems. Two-class and multi-class cases are studied separately with different learning challenges and focuses. (chapters 5 and 6)

- The first systematic study of the "multi-class" difficulties in class imbalance learning with separate and in-depth discussions of "multi-minority" and "multi-majority" cases. (chapter 6)

- A thorough empirical study on a set of software engineering problems, which are highly imbalanced in nature and come from real projects. (chapter 5)

1.5 Thesis Structure

1.5.1 Thesis Structure

This chapter presented a general picture for the whole thesis, clarified our research questions, and summarized the main contributions.

In chapter 2, we review the literature relating to "class imbalance learning" and "diversity in classification ensembles". Section 2.1 introduces the most commonly used techniques of handling class imbalance problems and corresponding evaluation criteria in two-class learning scenarios. Section 2.2 describes the current research progress in multi-class imbalance learning. Section 2.3 reviews existing diversity studies in classification ensembles, including the definitions of various diversity measures and their role in generalization. Section 2.4 reviews main negative correlation learning algorithms proposed so far.

In chapter 3, we give a systematic analysis of ensemble diversity for class imbalance learning, to answer the research questions in section 1.3.1. We build mathematical links between Q-statistic, a widely used diversity measure for classification ensembles, and several single-class performance measures. Six different situations are found based on the pattern analysis, showing how the single-class performance is affected by diversity. Then, they are verified in imbalanced scenarios empirically, where we proceed with comprehensive correlation analysis on artificial and real-world data sets with highly skewed class distributions. The relationship between diversity and overall performance is also discussed. Strong correlations are found between diversity and the chosen performance measures. Diversity shows a positive impact on the recognition of the minority class and benefits the overall performance of ensembles.

In chapter 4, we propose a new learning algorithm based on the idea of negative correlation learning, named "AdaBoost.NC", in order to address the issues in section 1.3.3. An "ambiguity" term decomposed from the 0-1 error function is introduced into the training framework of AdaBoost. We give both theoretical and empirical evidences to support its effectiveness in general cases.

In chapter 5, we explore the learning ability of AdaBoost.NC in depth to solve two-class imbalance problems, following the suggested conclusions in chapters 3 and 4. In order to answer the research questions in section 1.3.2, comprehensive experiments are conducted to evaluate AdaBoost.NC and other NCL algorithms, in comparison with state-of-art ensemble solutions for class imbalance problems. AdaBoost.NC is shown to perform the best in AUC and minority-class performance when working with random oversampling.

In chapter 6, the study in chapter 5 is extended to "multi-class" cases, where multiple minority or/and majority classes exist. We first investigate the impact of "multi-class" on the classification of class imbalance problems and point out the potential issues. Based on the findings, we then use the best ensemble model obtained in chapter 5, oversampling+AdaBoost.NC, to solve a set of real-world imbalanced problems with more than two classes, in comparison with existing state-of-art approaches. Its generalization ability is explored and exploited in a wider scope. It keeps the good capability of recognizing minority class examples and better balances the performance among classes in terms of G-mean, which is one of the most frequently used measures of overall performance in class imbalance learning.

In chapter 7, we conclude this thesis, summarize the contributions to ensemble learning and class imbalance learning areas, and propose several future studies suggested by our research so far.

1.5.2 Publications Resulting from the Thesis

The work resulting from these investigations has been reported in the following publications:

Submitted journal publications:

[1] S. Wang and X. Yao. Relationships Between Diversity of Classification Ensembles and Single-Class Performance Measures. In *IEEE Transactions on Knowledge and*

Data Engineering, 2011.

[2] S. Wang, L. L. Minku, and X. Yao. Negative Correlation Learning for Class Imbalance Problems. In *IEEE Transactions on Knowledge and Data Engineering*, 2011.

[3] S. Wang and X. Yao. Multi-Class Imbalance Problems: Analysis and Potential Solutions. In *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 2011.

Refereed conference publications:

[1] S. Wang and X. Yao. The Effectiveness of A New Negative Correlation Learning Algorithm for Classification Ensembles. In *IEEE International Conference on Data Mining Workshops 2010*, pages 1013-1020, 2010.

[2] S. Wang, H. Chen, and X. Yao. Negative correlation learning for classification ensembles. In *International Joint Conference on Neural Networks*, WCCI, IEEE Press, pages 2893-2900, 2010. (Travel Grant Awarded)

[3] S. Wang and X. Yao. Theoretical study of the relationship between diversity and single-class measures for class imbalance learning. In *IEEE International Conference on Data Mining Workshops 2009*, pages 76-81, 2009.

[4] S. Wang, K. Tang, and X. Yao. Diversity exploration and negative correlation learning on imbalanced data sets. In *International Joint Conference on Neural Networks 2009*, IEEE Press, pages 3259-3266, 2009. (Travel Grant Awarded)

[5] S. Wang and X. Yao. Diversity analysis on imbalanced data sets by using ensemble models. In *IEEE Symposium on Computational Intelligence and Data Mining 2009*, pages 324-331, 2009.

CHAPTER 2

LITERATURE REVIEW

This chapter gives the background knowledge and reviews the relevant literature to this thesis, including the reasons why ensemble learning approaches are widely used in class imbalance learning and the main issues encountered when an ensemble system is created for class imbalance problems. Section 2.1 reviews existing ensemble methods and major techniques in class imbalance learning, from which our main research questions are identified. Section 2.2 reviews proposed methods for learning from imbalanced problems in the presence of multiple minority or/and majority classes. For the consideration that class imbalance learning is a specific type of classification problems, section 2.3 describes current studies of diversity in classification ensembles. Section 2.4 explains the idea of negative correlation learning (NCL) as an important technique of ensemble learning that encourages diversity explicitly, and introduces main algorithms under this topic. Section 2.5 summarizes this chapter.

2.1 Learning from Class Imbalance Problems

Many conventional learning algorithms are not appropriate for learning from data with imbalanced class distributions. Their training procedure with the aim of maximizing overall accuracy often leads to a high probability of the induced classifier producing the majority-class label and a very low recognition rate for the minority class. It is useless for future prediction, considering that the minority class usually has a much higher misclassification cost and deserves better performance. The negative effect of class imbalance on classifiers, such as decision trees (He and Garcia, 2009; Japkowicz and Stephen, 2002), neural networks (Visa and Ralescu, 2005; Japkowicz and Stephen, 2002), k-Nearest Neighbour (kNN) (Kubat and Matwin, 1997; Batista et al., 2004; Zhang and Mani, 2003) and SVM (Yan et al., 2003; Wu and Chang, 2003), has been reported in the literature.

For a better understanding, we give some concrete descriptions here. 1-NN is a method that labels an example with the class belonging to its nearest neighbour. If the imbalance rate is very high, the nearest neighbour of a minority class example is very likely to be an example from the majority class. A misclassification thus happens. Constructing a decision tree is a recursive and top-down greedy search procedure (Quinlan, 1986). It selects the feature that can best split the classes of examples at each node. As a result, the training set is successively divided into smaller subsets, corresponding to disjoint rules for each class. However, the majority class dominates the leaves, and successive partitioning results in very specific rules for the minority class. Some papers specialized in the performance analysis of decision trees in the presence of imbalanced data (Chawla, 2003; Weiss and Provost, 2003; Drummond and Holte, 2003).

Numerous methods have been proposed to adjust the bias towards the majority class at data and algorithm levels. Data-level methods refer to a variety of resampling techniques, manipulating training data to rectify the skewed class distributions. They are simple and efficient, but their effectiveness depends on the solving problem greatly. How to tune them properly is a key issue. Various algorithm-level methods address class imbalance by modifying their training mechanism with the more direct goal of better accuracy on the minority class, including one-class learning (Japkowicz et al., 1995) and cost-sensitive learning algorithms. Rather than differentiating examples of one class from the other, one-class learning (also known as novelty detection and recognition-based methodology (Chawla et al., 2004)) learns a model by using mainly or only a single class of examples alone. One attempts to measure the similarity for a new input to the learnt class, so as to judge whether it belongs to this class. Some representative works from this category include one-class SVM (Raskutti and AdamKowalczyk, 2004; Manevitz and Yousef, 2001; Scholkopf et al., 2001; Zhuang and Dai, 2006) and autoencoder methods (Manevitz and Yousef, 2007; Japkowicz, 2001b; Japkowicz et al., 1995). One-class learning was shown to be useful in dealing with extremely imbalanced data sets with high dimensional feature space (Raskutti and AdamKowalczyk, 2004; Lee and Cho, 2006). With respect to cost-sensitive methods, decision trees and neural networks have been made capable of processing different misclassification costs of classes, such as adjusting the decision threshold (Japkowicz, 2000b), making the tree split criteria cost-sensitive (Drummond and Holte, 2000; Ting, 2002), and applying cost-sensitive modifications to the error minimization function or outputs of the neural network (Kukar and Kononenko, 1998; Zhou and Liu, 2006b; Alejo et al., 2009). A cost-sensitive learning method is preferred only when clear misclassification costs of classes are available. In many situations, however, this information is unknown. Besides, algorithm-level solutions require different treatments for specific kinds of learning algorithms. It restricts their use in many applications, since we do not know which algorithm would be the best choice beforehand in most cases. A good solution is expected to be able to deal with most given problems effectively (Visa and Ralescu, 2005). Therefore, it is more meaningful to explore solutions that are applicable to a wider scope as a major aim of this thesis.

In addition to the aforementioned data-level and algorithm-level solutions, ensemble learning has become another major category of approaches to handling class imbalance problems. From the individual level, it can be easily adapted for emphasizing the minority class by integrating different resampling techniques (Li, 2007; Liu et al., 2009; Chawla et al., 2003) or by applying different misclassification costs (Joshi et al., 2001; Chawla and Sylvester, 2007; Guo and Viktor, 2004; Fan et al., 1999). From the ensemble level, the idea of combining multiple classifiers can make up every single classifier's weaknesses and lower down the risk of overfitting by reducing the error variance (Brown et al., 2005). Therefore, an ensemble is believed to outperform a single classifier with more stable performance by learning from different views.

Generally speaking, great research efforts of solutions for class imbalance learning in the literature have been made in resampling techniques, one-class learning algorithms, cost-sensitive learning methods and ensemble learning methods. In order to establish a less specific, simple and effective way to deal with class imbalance problems, we put our focus on resampling techniques and ensemble learning methods for their good flexibility and effectiveness, which will be introduced in this section in more detail. Important motivations of this thesis are discovered, based on which our attention in this thesis will be devoted to a deeper understanding and the improvement of existing ensemble methods in class imbalance learning. How to evaluate class imbalance learning algorithms is a crucial issue. A review of major evaluation criteria will be provided following the algorithm introduction.

2.1.1 Resampling Techniques

Resampling is a group of data-level techniques that adjust the distribution of training data directly. It includes oversampling and undersampling methods. Oversampling increases the number of minority class examples until the imbalance is eliminated. Likewise, undersampling removes some examples from the majority class until the data set is relatively balanced. There are various ways of enlarging the minority class and shrinking the majority class. We give them respective descriptions next. For a training data set Z with N examples taken by the algorithm as input ($Z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, cardinality $|Z| = N$), each x_i belongs to an instance space X, and each label y_i belongs to a finite label set $Y = \{\omega_1, \ldots, \omega_c\}$. Particularly, $c = 2$ represents a two-class problem with one minority-class subset $Z_{min} \subset Z$ and one majority-class subset $Z_{maj} \subset Z$. Without loss of generality, we have $|Z_{min}| < |Z_{maj}|$, $Z_{min} \cap Z_{maj} = \emptyset$ and $Z_{min} \cup Z_{maj} = Z$ hold. The obtained data set after resampling is denoted by S, composed of disjoint subsets S_{min} containing all minority class examples and S_{maj} containing all majority class examples.

Oversampling

(1) Random oversampling (abbr. ROS)

It is a non-heuristic method that makes replicates of randomly selected examples in Z_{min} and adds them to S. The degree of class distribution can be adjusted to any desired level. It is simple, but it has been argued that exact copies can lead to overfitting (Chawla et al., 2002; Batista et al., 2004; Weiss and Provost, 2001).

(2) Synthetic Minority Over-sampling TEchnique (abbr. SMOTE) (Chawla et al., 2002)

It is an advanced oversampling technique that has shown great success in many applications. It generates new data points labelled as minority based on the similarities between original minority class examples in the feature space. Concretely, to create a synthetic example, SMOTE randomly selects one of the k-nearest neighbours in Z_{min} of a minority example x_i, denoted by $\hat{x}_i$, calculates the difference between them, and then adds the difference vector where each dimension is multiplied by a random number δ in range $[0, 1]$ to x_i. It is a point along the line between x_i and $\hat{x}_i$. By doing so for certain rounds, it forces the decision boundaries of the minority class to spread towards the majority class region effectively. The pseudo-code for SMOTE is given in Table 2.1. The oversampling rate M and the number of nearest neighbours k are pre-defined parameters.

SMOTE is reported to have the over-generalization problem (Wang and Japkowicz, 2004). It does not consider the neighbourhood with regard to the majority class, thus resulting in the possibility of overlapping between classes (Prati et al., 2004).

(3) Borderline-SMOTE (Han et al., 2005)

Borderline-SMOTE is a modification of SMOTE. It assumes that the examples near classification boundaries (i.e. borderline examples) are more likely to be misclassified and thus more important. Only borderline examples are used to generate new data by applying SMOTE. If most nearest neighbours of a minority class example belong to the majority class, it is treated as a borderline example in “danger”.

Table 2.1: The SMOTE algorithm (Chawla et al., 2002).

```
Given: the SMOTE rate $M$; number of nearest neighbour $k$.
Let: $S_{min} = Z_{min}$.

For each minority example $x_i$ ($i = 1, \ldots, |Z_{min}|$):
Step 1. Compute the k-NN set $\{\hat{x}_i\}$ for $x_i$ in $Z_{min}$.
Step 2. For j = 1 to $M$:
        2.1. Choose a random neighbour $\hat{x}_i \in \{\hat{x}_i\}$.
        2.2. Compute difference vector $dif = \hat{x}_i - x_i$.
        2.3. Multiply the value on each dimension of $dif$ by a random number
             $\delta \in [0, 1]$.
        2.4. $x_{new} = x_i + dif$.
        2.5. Add $x_{new}$ to $S_{min}$.

Return: $S_{min}$.
```

(4) Adaptive Synthetic Sampling (abbr. ADASYN) (He et al., 2008)

Its main idea is to generate minority class examples adaptively according to their distributions, where more synthetic data is created for "difficult" ones than "easy" ones. Similar to Borderline-SMOTE, "difficult/easy" is determined by the neighbourhood of a minority class example. For each x_i in Z_{min}, ADASYN calculates the proportion of majority class examples in the k-NN set of x_i, denoted by Δ_i. Normalize Δ_i so that $\sum \Delta_i = 1$. Then, the number of synthetic examples that need to be generated for x_i is $\Delta_i \times T$, where T is the desired amount of new data (i.e. $|S_{min}| - |Z_{min}|$). Δ describes a density distribution that decides the number of synthetic examples automatically.

Undersampling

(1) Random undersampling (abbr. RUS)

It is a non-heuristic method that removes data from Z_{maj} randomly until the minority and majority classes have a comparable size. It enforces balance with a radical effect, and may discard useful information pertaining to the majority class for better learning (Batista et al., 2004; He and Garcia, 2009).

(2) Tomek links (Tomek, 1976)

Given two examples x_i in Z_{min} and x_j in Z_{maj}, $d(x_i, x_j)$ stands for the distance between them. The (x_i, x_j) pair is called a Tomek link if there does not exist any example x_k, such that $d(x_i, x_k) < d(x_i, x_j)$ or $d(x_j, x_k) < d(x_i, x_j)$. Either one in the link is noise, or both are near the boundary. As an undersampling method, only the ones belonging to the majority class in the links are removed. It can clean up overlapping between classes and establish well-defined classification rules. Finding Tomek links could be computationally expensive (Batista et al., 2004).

(3) Condensed nearest neighbour rule (abbr. CNN) (Hart, 1968) and One-sided selection (abbr. OSS) (Kubat and Matwin, 1997)

CNN is a data cleaning technique that aims to find examples far from decision boundaries. The idea is to find a consistent data subset $\hat{Z} \subseteq Z$, where all the examples in Z can be classified correctly by using 1-NN in $\hat{Z}$. OSS uses it to find and remove redundant examples in Z_{maj} first, which are outside the consistent subset. Then, the Tomek links method is applied to the obtained consistent subset, so as to pick out borderline and noisy examples from the majority class.

(4) Wilson's edited nearest neighbour rule (abbr. ENN) (Wilson, 1972) and Neighbourhood cleaning rule (abbr. NCR) (Jorma, 2001)

ENN removes any example whose class label differs from the class of at least two of its three nearest neighbours. Based on the idea, NCR removes the majority class examples that are misclassified by 3-NN, which are considered as noisy. Meanwhile, if a minority class example is misclassified by 3-NN, its neighbours that belong to the majority class are removed.

Quite a few papers have discussed the effectiveness of resampling techniques and their combinations. Generally speaking, resampling techniques have shown improved classification performance for most imbalanced data sets (Seiffert et al., 2008b; Japkowicz, 2000a). They work independently with learning algorithms, and are thus more versatile than algorithm-level methods. However, it is always an issue of tuning them effectively. It is unclear which resampling method performs better and which sampling rate should be used. Some empirical studies found that resampling to full balance is not necessarily optimal, and the best resampling rate varies with problem domains and resampling techniques (Estabrooks et al., 2004). Their performance also depends on different classifiers and evaluated measures (Hulse et al., 2007). It is difficult to choose the "right" one when an imbalanced task is given without much prior data knowledge.

2.1.2 Ensemble Learning Methods

Due to the lack of guidance for the choice of resampling techniques, the idea of combining multiple classifiers arose, in order to make up each other's weaknesses (Kotsiantis et al., 2006; Kotsiantis and Pintelas, 2003; Estabrooks et al., 2004; Seiffert et al., 2008a,b). Ensemble learning thus has become one of the major techniques in class imbalance learning. It allows individual classifiers to emphasize the minority class regions differently, and exploits their combination to lower down the risk of overfitting for better generalization. Many ensemble methods were proposed with varying degrees of success. Existing ensemble solutions so far mainly focus on how to rebalance the training subset for each base classifier from the data level and how to make the ensemble cost-sensitive. In the following, we will first describe the ineffectiveness of some traditional ensemble algorithms in dealing with class imbalance problems, and then explain why they are still exploited widely in various ways by reviewing some frequently used solutions.

Bagging (Breiman, 1996) and AdaBoost (Freund and Schapire, 1996) are two most popular ensemble learning techniques in the literature (Brown, 2010; Bauer and Kohavi, 1999). Especially, AdaBoost is reported to have greater benefits to the prediction accuracy of a weak learning algorithm (Quinlan, 1996). In class imbalance learning, however, both become less effective in recognizing rare cases (Liu et al., 2009; Valdovinos and Sanchez, 2005).

Bagging-Based Methods

The Bagging algorithm (Breiman, 1996), the acronym of Bootstrap AGGregatING, constructs multiple base classifiers by sampling the training examples at random with replacement from data space, as known as *bootstrapping*. The final prediction is uniformly voted by those sub-classifiers. Apparently, applying the original Bagging algorithm to classifying an imbalanced data set is not a wise idea. Every training subset bootstrapped from the original data still has an imbalanced distribution. It could be biased even more towards the majority class when compared with the original training set (Zhu, 2007). Either ignoring or overfitting the minority class examples is very likely to happen to each base classifier (He and Garcia, 2009). Consequently, the final performance will favor the majority class. Specific techniques are necessary to compensate this situation. A straightforward way is to correct this skewness within each subset by resampling, and can thus build component classifiers from data with balanced class distributions (Barandela et al., 2003).

Some representatives of Bagging-based methods include Tao et al.'s AB-SVM (Tao et al., 2006), Li's BEV (abbr. of Bagging Ensemble Variation) (Li, 2007), Chan et al.'s combining model (Chan and Stolfo, 1998), and Yan et al.'s SVM ensemble (Yan et al., 2003). They undersample the majority class examples into several subsets with the equal size of the minority class, each of which is then merged with the entire minority class examples. Every classifier is thus trained from a strictly balanced subset. AB-SVM simply bootstraps examples from the majority class, while the others divide the majority class examples into disjoint sets. However, it has been argued that these Bagging variations inherit the disadvantage of sacrificing too much performance on the majority class from undersampling. Especially when data is highly imbalanced, it is difficult to obtain a representative subset by using undersampling alone. Besides, since the same minority class examples are taken as input by every classifier, the resulting ensemble still have a high possibility of suffering from overfitting (Liu et al., 2009). Other techniques are usually required to make up the drawbacks.

Liu et al. proposed two ensemble methods to strengthen the use of majority class examples, called EasyEnsemble and BalanceCascade (Liu et al., 2009). They construct an ensemble of ensembles, where each individual classifier is also an ensemble of AdaBoost (Freund and Schapire, 1996), but use different sampling strategies. Similar to AB-SVM, EasyEnsemble samples majority class examples randomly. Selected ones are then combined with all minority class examples for forming training subsets. Each subset in EasyEnsemble is believed to be better learnt through AdaBoost. EasyEnsemble benefits from the combination of Boosting and the Bagging-like sampling strategy with balanced class distribution. BalanceCascade explores the majority class in a supervised manner by paying more attention to the misclassified ones. It builds classifiers sequentially, during which some correctly classified majority class examples are removed from the training set based on the conjecture that they are somewhat "redundant".

Boosting-Based Methods

In addition to Bagging-based approaches, Boosting-based methods appear to be more popular. AdaBoost (Freund and Schapire, 1996) is the most well-known algorithm in Boosting family (Schapire, 2002). It builds classifiers sequentially with subsequent classifiers focusing on training examples that are misclassified by earlier ones. A set of weights is maintained over the training set, which are updated at the end of each round according to the accuracy of the current classifier, and then fed back into the next round. It is observed to be capable of both bias and variance reduction, but does not deal well with noise (Bauer and Kohavi, 1999; Quinlan, 1996).

The emphasis on misclassified examples makes AdaBoost an accuracy-oriented algorithm. Classifiers having higher accuracies receive higher weights. It treats all classes equally. In the presence of imbalanced data, its learning strategy tends to bias towards the majority class as rare examples contribute less to the overall classification accuracy (Joshi et al., 2002; Sun et al., 2007). Therefore, the original AdaBoost is not good at recognizing rare cases. In addition, it may suffer from the overfitting problem, for the reason that rare examples tend to be weighted more than common ones, and thus replicated and boosted more often (Weiss, 2004; Kotsiantis et al., 2006; Guo et al., 2008).

Nevertheless, AdaBoost is still popular in the field of class imbalance learning, for the reason that it can be easily adapted for advancing the classification performance on the small class with following merits (Weiss, 2004; Kotsiantis et al., 2006). First, it is applicable to most classification algorithms (flexibility). Second, the optimal class distribution and representative examples are explored automatically without extra computational cost (efficiency). Third, there is no need to eliminate any majority class examples (accuracy) (Sun et al., 2007). These attractive properties encourage AdaBoost to have evolved in two ways – the integration with resampling techniques (Seiffert et al., 2008b) and cost-sensitive Boosting (He and Garcia, 2009).

When it is integrated with resampling techniques, data generation methods are often used to broaden the decision region for the minority class. For instance, SMOTEBoost (Chawla et al., 2003) introduces SMOTE (Chawla et al., 2002) in each round of Boosting, to enhance the probability of selecting the difficult rare cases that is dominated by the majority class examples. Meanwhile, Boosting prevents from sacrificing accuracy over the entire data set. It is believed to produce higher diversity within the ensemble. It shows improved classification performance compared to the conventional Boosting, but the parameter setting in SMOTE is crucial, which could cause over-generalization. Besides, generating new data could be computationally costly (He and Garcia, 2009). Table 2.2 describes the SMOTEBoost algorithm following the AdaBoost.M2 procedure (Freund and Schapire, 1996).

Mease et al. proposed a faster method, JOUS-Boost (Mease et al., 2007). Instead of generating new examples for the minority class, it simply adds a small amount of random

Table 2.2: The SMOTEBoost algorithm (Chawla et al., 2003).

```
Given: a training set $Z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$; the SMOTE rate $M$;
number of nearest neighbour $k$.
Let: $B = \{(i, y) : i = 1, \ldots, N, y \neq y_i\}$.
Initialize data weights $D_1(i, y) = 1/|B|$ for $(i, y) \in B$.

For each training epoch $j = 1, 2, \ldots, L$:
Step 1. Apply SMOTE(M,k) to the minority classes, and obtain a synthetic
        data set $\{x_{new}\}$.
Step 2. Train a weak classifier $f_j$ using selected examples based on $D_j$ and $\{(x_{new}, y)\}$:
        $X \times Y \rightarrow [0, 1]$.
Step 3. Compute the pseudo-loss of $f_j$:
        $\varepsilon_j = \sum_{(i,y) \in B} D_j(i, y)(1 - f_j(x_i, y_i) + f_j(x_i, y))$.
Step 4. Set $\beta_j = \epsilon_j / (1 - \epsilon_j)$.
Step 5. Update weights $D_j$: $D_{j+1} = (D_j / Z_j)\, \beta_j^{(1/2)(1 - f_j(x_i, y) + f_j(x_i, y_i))}$,
        where $Z_j$ is a normalization factor.

Output the final ensemble: $H(x) = \arg \underset{y \in Y}{max} \sum_{j=1}^{L} \left(\log \frac{1}{\beta_j}\right) f_j(x, y)$.
```

noise ("jittering") to the replicates caused by oversampling in each round of Boosting. It shows efficient results in experiments, but the over-generalization problem still exists due to the introduction of "noise".

Cost-sensitive Boosting is another class of ensemble solutions for class imbalance learning. The key idea is to increase the probability of selecting costly examples in each round by manipulating the weight updating rule. They aim at minimizing the total misclassification cost instead of the classification error, such as AdaCost (Fan et al., 1999), CSB1 and CSB2 (Ting, 2000). A cost matrix or associated cost items must be specified prior to learning. However, it is hard to obtain an explicit cost description in many situations. To overcome this issue, RareBoost (Joshi et al., 2001) was proposed as a cost-sensitive Boosting method. Its attractive feature is that the cost is determined dynamically by the performance of the current classifier based on the proportions of four types of examples TP (true positives), FP (false positives), TN (true negatives) and FN (false negatives). The constraint of $TP > FP$ and $TN > FN$ is necessary. Sometimes, it is a quite strong condition for the minority class. Existing cost-sensitive Boosting algorithms have been categorized into three forms by Sun et al. (Sun et al., 2007). Since no new data information is introduced into training, the learning algorithm might over-emphasize the minority class with a higher cost assigned. Cost-sensitive Boosting could still confront the problem of overfitting minority class examples.

Generally speaking, ensemble methods attempt to make use of the difference of individual classifiers for the performance improvement by adjusting the learning focus on the minority class differently during training (Estabrooks et al., 2004; Kotsiantis and Pintelas, 2003; Chawla and Sylvester, 2007). However, no study has actually shown the effects of this difference in classifying imbalanced data sets to date, which motivates us to take this step further in this thesis.

2.1.3 Evaluation Criteria

For a traditional classification task, overall accuracy is the most commonly used criterion of performance evaluation. As we have described in the Introduction chapter, it is not appropriate for class imbalance problems and can provide misleading conclusions, since it is strongly biased to favor the majority class. A classifier can achieve very high accuracy but performs poorly in predicting examples from small classes. Overall accuracy becomes meaningless when the learning concern is how to find rare examples effectively. Therefore, other performance criteria have been adopted under this scenario. This section introduces the major evaluation measures for class imbalance learning.

For a two-class problem, classification performance can be represented by a confusion matrix, as illustrated in Table 2.3.

Table 2.3: Confusion matrix.

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

A classifier produces four types of examples on testing data,

- TP: the number of correctly classified examples belonging to the positive class.
- TN: the number of correctly classified examples belonging to the negative class.
- FP: the number of misclassified examples belonging to the negative class.
- FN: the number of misclassified examples belonging to the positive class.

By convention, we treat the minority class as positive and the majority class as negative. Based on the four metrics, following measures are defined and have been frequently used in the class imbalance learning literature, which can be divided into two groups: single-class performance measures that evaluate how well a classifier performs in one class, particularly the minority class; overall performance measures that show how well a classifier can balance the performance among classes.

Single-Class Performance Measures

Recall (R), precision (P) and F-measure (F) come from information retrieval area (Rijsbergen, 1979) and are adopted to assess single-class performance in classifier evaluation of class imbalance problems. For the positive class, they are defined as

$$R = \frac{TP}{TP + FN}, \tag{2.1}$$

$$P = \frac{TP}{TP + FP}, \tag{2.2}$$

$$F = \frac{(1 + \rho^2) \cdot R \cdot P}{\rho^2 \cdot R + P}, \tag{2.3}$$

where ρ controls the relative importance of recall and precision. It is usually set to 1.

Recall is a measure of completeness – the proportion of positive class examples that are classified correctly to all positive class examples. Precision is a measure of exactness – the proportion of positive class examples that are classified correctly to the examples predicted as positive by the classifier (He and Garcia, 2009). F-measure incorporates both recall and precision to express the trade-off between them. It was shown to be a more favorable measure (Joshi, 2002) and has been used as a criterion for classifier selection (Chawla and Sylvester, 2007; Sun et al., 2006).

Overall Performance Measures

Kubat et al. (Kubat et al., 1997) proposed to use G-mean to replace overall accuracy. It is the geometric mean of recall values of the positive and negative classes. A good classifier should have high accuracies on both classes, and thus a high G-mean.

ROC analysis (abbr. of Receiver Operating Characteristic) (Fawcett, 2003, 2006) and the associated use of the area under the ROC curve (i.e. AUC) (Bradley, 1997; Ling et al., 2003) are probably the most common technique to evaluate overall classification performance in this area. They measure the separating ability of a classifier between two classes. A ROC curve depicts all possible trade-offs between TP rate and FP rate, which are defined as

$$\text{TP rate} = \frac{TP}{TP+FN}; \text{ FP rate} = \frac{FP}{FP+TN}.$$

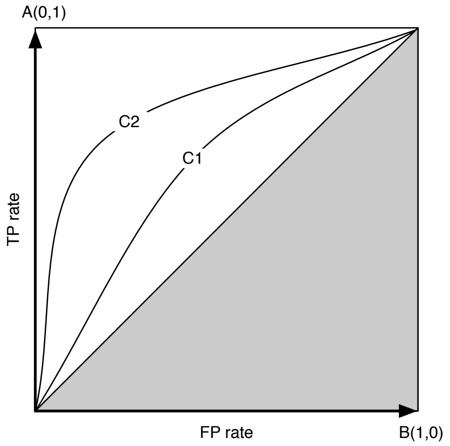


Figure 2.1: Illustration of ROC curves (He and Garcia, 2009).

TP rate is equivalent to the recall measure of the positive class. TP rate and FP rate can be understood as the benefits and costs of classification with respect to data distributions. Each point on the curve corresponds to a single trade-off. A better classifier should produce a ROC curve closer to the top left corner. As illustrated in Fig. 2.1, curves C1 and C2 represent two ROC curves. The corresponding classifier associated with C2 can provide better performance compared to the classifier associated with C1. Point A (0,1) represents a perfect classification. The diagonal line represents the situation of a classifier randomly guessing the class labels. Therefore, any classifier that falls into the shaded area of the ROC space (i.e. the lower right triangle) performs worse than random guessing. A ROC curve of a classifier is often generated by varying the classification decision threshold for separating positive and negative classes (Monard and Batista, 2002; Maloof, 2003; Fawcett, 2003, 2006), which produces a set of (TP rate, FP rate) points.

Closely related to ROC, AUC represents a ROC curve as a single scalar value by estimating the area under the curve, varying in $[0, 1]$. A random guess has an AUC value of 0.5. According to Hand and Till (Hand and Till, 2001), AUC is equivalent to the probability that a randomly chosen example of the positive class will have a smaller estimated probability of belonging to the negative class than a randomly chosen example of the negative class. It is proved to be independent of the selected decision threshold and class distributions (Fawcett, 2003). It should be noted that the ROC analysis and AUC estimation are only able to describe the discriminability of a pair of classes. The extension of AUC for multi-class evaluation will be given in section 2.2.3.

2.2 Learning from Multi-Class Imbalance Problems

Most efforts of class imbalance learning so far have been made to handle two-class imbalance problems. A great number of real-world applications, however, have more than two classes with imbalanced distributions. They pose new challenges that are not observed in two-class problems. Multi-class imbalance problems have been shown to suffer from more classification difficulties. Zhou and Liu (Zhou and Liu, 2006b) examined the effectiveness of various class imbalance learning techniques developed for two-class problems on a set of cost-sensitive data sets with multiple classes. Their experimental results showed that most of the techniques become ineffective and may even cause negative effects to multi-class tasks. Dealing with multi-class tasks is generally more difficult. Without much work done on this topic so far, more investigations into multi-class imbalance problems are necessary to explain what problems multi-class can cause to existing class imbalance learning techniques and how it affects the classification performance.

Among the limited works of dealing with multi-class imbalance problems, most methods seek help from class decomposition, which breaks down the whole problem into a series of small two-class sub-problems. For each sub-problem, two-class techniques are applied to addressing class imbalance. In this section, we first introduce different types of class decomposition schemes. Then, we review current solutions and performance evaluation criteria for multi-class imbalance learning.

2.2.1 Class Decomposition

Class decomposition is a technique of processing multi-class by transforming the problem into multiple two-class classification problems. It has been categorized into three types (Ou and Murphey, 2007): *one-against-all* (OAA, also known as one-vs-others, one-vs-rest), *one-against-one* (OAO) and *P-against-Q* (PAQ). Decomposing a big problem has following advantages:

- Individual classifiers are likely to be simpler than a classifier learnt from the whole data set.
- They can be trained simultaneously for less modeling time.
- They can be trained independently, which allows different feature spaces, feature dimensions and architectures. The change of one classifier will not affect the others.

Its potential drawback is that each individual classifier is trained without full data knowledge. It can cause classification ambiguity or uncovered data regions with respect to each type of decomposition. Given a c-class task ($c > 2$),

(1) OAA scheme (Rifkin and Klautau, 2004)

Each of the c classes is trained against all other classes. It results in c binary classifiers. The final decision will be the class label whose corresponding binary classifier produces the highest output value among all. However, it can make training data highly imbalanced (Tan et al., 2003). If the original data is imbalanced already or contains a large number of classes, OAA will make the problem even worse. In addition, the classification boundary produced by the individual classifier is independent from each other, which can lead to uncovered and overlapped regions in the data space. It is unable to exploit the fact that each example belongs to only one class (Jin and Zhang, 2007; Valizadegan et al., 2008).

(2) OAO scheme (Hastie and Tibshirani, 1998)

Each of the c classes is trained against every one of the other classes. It results in $c\,(c-1)\,/2$ binary classifiers. A combining strategy of their outputs is necessary for a final decision. The simplest way is a majority vote, which outputs the class label with the most votes. Major advantages of OAO are,

- It provides redundancy to the prediction of each class, which can be beneficial to generalization performance. If one classifier makes a classification mistake, others still have chance to make it up.
- It does not produce imbalanced training data.
- It is capable of incremental learning. When a new class joins to the current data, we just need to build another c new classifiers without affecting the existing ones.

It also suffers from some known disadvantages (Tan et al., 2003),

- The number of individual classifiers grows fast in a quadratic rate of c. When c is large, the training time can be very long.

- Due to different learning problems for each classifier, combining their results can cause potential classification errors.

(3) PAQ scheme (Ou and Murphey, 2007)

Using P of the c classes against the other Q of the c classes, the training process is repeated several times. Different P classes are chosen at each time. Which classes to choose is decided by a codeword table with 0/1 bits. There are several methods to produce a codeword table. Error-correcting output code (ECOC) (Dietterich and Bakiri, 1995) is one of the most well-known and popular methods. The major disadvantage of PAQ scheme is that its performance depends on many factors, including base classifiers and the codeword table (Jin and Zhang, 2007).

2.2.2 Existing Solutions

Most existing solutions for multi-class imbalance problems use class decomposition schemes to handle multi-class and work with two-class imbalance techniques to handle each obtained binary sub-task. For example, protein classification is a typical multi-class imbalance problem. Tan et al. used both OAA and OAO schemes with rule-based learners to improve the coverage of examples from small protein classes (Tan et al., 2003). Zhao et al. used OAA to handle multi-class, and undersampling and SMOTE (Chawla et al., 2002) to compensate the imbalanced distribution in their protein data (Zhao et al., 2008). Liao investigated a variety of oversampling and undersampling techniques used with OAA for a weld flaw classification problem (Liao, 2008). Chen et al. proposed an algorithm, using OAA to deal with multi-class and then applying some advanced sampling methods that can further decompose each binary problem so as to rebalance the training set (Chen et al., 2006). Fernandez integrated OAO with SMOTE in their algorithm (Fernández et al., 2010). Instead of using data-level methods, Alejo et al.'s algorithm made the error function of neural networks cost-sensitive by incorporating the proportion of classes within the data set to emphasize minority classes, after OAA was applied (Alejo et al., 2009).

Different from the above methods, a cost-sensitive ensemble algorithm was proposed (Sun et al., 2006), which addressed multi-class imbalance directly without using class decomposition. The key points of this algorithm are to find an appropriate cost matrix with multiple classes and introduce the costs into the algorithm. They applied a genetic algorithm (GA) to searching the optimal cost setup of each class. Two kinds of fitness were tested, G-mean and F-measure, whose choice depended on the training objective of the given problem. The obtained cost vector was then integrated into a cost-sensitive version of the Adaboost.M1 algorithm (Freund and Schapire, 1997), named AdaC2 (Sun et al., 2005, 2007), which is able to process multi-class data sets. However, searching the cost vector is very time-consuming due to the nature of GA. There is no existing method that can deal with multi-class imbalance problems efficiently and effectively yet. Chapter 6 of this thesis will first analyze why multi-class makes a problem harder and then explore new approaches to tackling the difficulties.

2.2.3 Evaluation Criteria in the Presence of Multi-Class

We now turn our attention to the assessment metrics. While all the single-class performance measures in section 2.1.3 are still suitable for multi-class problems, overall performance measures have to be extended.

Sun et al. (Sun et al., 2006) adapt G-mean to multi-class scenarios. It is defined as the geometric mean of recall values (R_i) of all classes. Given a c-class problem,

$$G - mean = \left(\prod_{i=1}^{c} R_i \right)^{1/c}. \tag{2.4}$$

It is able to capture the balanced performance among classes effectively, as the recognition rate of every class is equally taken into account.

A commonly accepted extension of AUC is proposed by Hand and Till (Hand and Till, 2001), called *M measure* or *MAUC*. It is the average of AUC of all pairs of classes, and defined as

$$M = \frac{2}{c\,(c-1)} \sum_{i<j} A\,(i,j)\,, \tag{2.5}$$

where $A\,(i,j) = [A\,(i|j) + A\,(j|i)]\,/2$ for class pair (i,j). $A\,(i,j)$ measures the separability between classes. $A\,(i|j)$ is the probability that a randomly drawn example of class j will have a lower estimated probability of belonging to class i than a randomly drawn example of class i. It should be noted that $AUC = A\,(i|j) = A\,(j|i)$ in the two-class scenario, but the equality does not hold when more than two classes exist. Although there are other modifications of AUC (Provost and Domingos, 2000), MAUC has the merit that it is insensitive to class distributions and error costs (Hand and Till, 2001; Fawcett, 2003, 2006; He and Garcia, 2009).

It is worth mentioning that, for the evaluation of learning algorithms based on class decomposition, some works chose to take the average of any two-class performance measure for produced binary classifiers (Zhao et al., 2008; Liao, 2008; He and Garcia, 2009). Finally, because of different importance and classifier performance among classes in class imbalance learning, single-class and overall performance measures are always considered simultaneously to evaluate different performance aspects and provide more insights into learning algorithms.

2.3 Diversity in Classification Ensembles

Diversity of ensembles has been a hot topic in ensemble learning during the past few years. It is commonly agreed that the success of ensembles is attributed to diversity – *the degree of disagreement within the ensemble* (Kuncheva et al., 2002; Brown et al., 2004). In the regression context, it has already been clearly quantified and measured in terms of covariance between individual learners by decomposing the ensemble error (Brown et al., 2005). In the classification context, the effect of diversity has also been recognized and exploited in algorithm design. For example, Bagging (Breiman, 1996) creates diversity by bootstrapping different training subsets; AdaBoost (Freund and Schapire, 1996) achieves diversity by altering the distribution of training examples for each classifier to avoid making same errors. Both algorithms have shown great success in the ensemble learning area. The underlying principle of creating a classification ensemble can be generally described as "*individual classifiers make errors on different examples*" (Polikar, 2006; Ali and Pazzani, 1995). However, there is no neat theory to rigorously define how differences between individual classifiers contribute to overall classification accuracy, depending on the type of error function and choice of combining methods. As such, it is hard to achieve an agreed definition of diversity for classification ensembles.

In spite of the resulting benefits of diversity in dealing with classification problems, some studies show counterintuitive empirical results due to its vague link to overall accuracy. For example, Kuncheva et al. found a weak relationship between diversity and overall accuracy through experimental discussions, and raised the doubt of the usefulness of diversity measures in training classification ensembles (Kuncheva and Whitaker, 2003). The similar observation also happened in Garcia et al.'s work (Garcia-Pedrajas et al., 2005). They proposed an evolutionary multi-objective method for designing ensembles. Their results showed that considering diversity as one of the objectives does not bring a clear performance improvement. Chen stated that diversity highly correlates with the classification error only when diversity is small, and its effect is reduced out of a specific range (Chen, 2008).

The partial understanding of diversity in classification ensembles leads to various metrics proposed to quantify diversity and continuous research efforts in the role of diversity in classification accuracy from different perspectives.

2.3.1 Diversity Measures for Classification

For quantitative assessment of diversity, several measures have been defined in two types, pairwise and non-pairwise. Pairwise measures calculate the average similarity or disagreement between all possible pairs of classifiers in the ensemble. This similarity/disagreement is expressed by a distance metric. Non-pairwise measures consider the voting distribution among classifiers and calculate the entropy or correlation. In particular, ten existing measures are frequently mentioned and studied in the majority of relevant literature (Kuncheva and Whitaker, 2003, 2001; Tang et al., 2006; Chen, 2008): Q-statistic (Q), correlation coefficient (ρ), disagreement measure (Dis), double-fault (DF), entropy (Ent), Kohavi-Wolpert variance (KW), interrator agreement (κ), difficulty measure (θ), generalized diversity (GD) and coincident failure diversity (CFD). The first four belong to pairwise measures, and the remaining ones are non-pairwise. All of them are based on the oracle output (correct/incorrect decision) and the combining method of majority vote. Their definitions will be given next.

Let $F = \{f_1, f_2, \ldots, f_L\}$ be a set of learnt classifiers as committee members of an ensemble, and $Y = \{\omega_1, \omega_2, \ldots, \omega_c\}$ be a finite label set. For any input x_j belonging to an instance space X, there is an expected label $y_j \in Y$. For each classifier f_i $(i = 1, \ldots, L)$, we define $O_{j,i} = 1$ if f_i recognizes x_j correctly, and 0 otherwise. Similarly for the combined model of those individuals, we define $O_{j,ens} = 1$ if the majority vote result is correct for example x_j, and 0 if x_j is misclassified. This is known as the *oracle type of output*. The measures based on the oracle output do not require any prior knowledge about data and any specific base learners. They are only determined by the correct/incorrect decisions of individuals. The oracle output provides a general model for analyzing various ensemble learning methods (Tang et al., 2006). Taking two classifiers f_i and f_k, we define N^{ab} as the number of examples x_j for which $O_{j,i} = a$ and $O_{j,k} = b$ within a labelled data set $Z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$. Table 2.4 presents the relationship between classifiers f_i and f_k. Each entry can also be in the form of probability by calculating N^{ab}/N.

(1) Q-statistic (Q) (Yule, 1900)

The Q-statistic for classifiers f_i and f_k is defined as

Table 2.4: A 2 × 2 table of the relationship between a pair of classifiers (Kuncheva and Whitaker, 2003).

	f_k correct (1)	f_k wrong (0)
f_i correct (1)	N^{11}	N^{10}
f_i wrong (0)	N^{01}	N^{00}

Total: $N = N^{00} + N^{01} + N^{10} + N^{11}$

$$Q_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}. \tag{2.6}$$

$Q_{i,k}$ assesses the similarity between them. For a set of L classifiers, diversity is measured by the averaged Q-statistic

$$Q = \frac{2}{L\,(L-1)} \sum_{i=1}^{L-1} \sum_{k=i+1}^{L} Q_{i,k}. \tag{2.7}$$

For statistically independent classifiers, the expectation of $Q_{i,k}$ is 0. It varies between -1 and 1. It performs positive if classifiers tend to recognize the same input examples correctly, and negative if they commit errors on different examples (Kuncheva and Whitaker, 2003). Larger Q-statistic values indicate smaller diversity. It is the most widely discussed diversity measure in the ensemble literature.

(2) Correlation coefficient (ρ) (Sneath and Sokal, 1973)

The correlation coefficient ρ is defined as

$$\rho_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{\sqrt{\left(N^{11}+N^{10}\right)\left(N^{01}+N^{00}\right)\left(N^{11}+N^{01}\right)\left(N^{10}+N^{00}\right)}}. \tag{2.8}$$

Q and ρ have the same sign, and $\rho \leq Q$. $\rho = 0$ indicates that the classifiers are uncorrelated.

(3) Disagreement measure (Dis) (Skalak, 1996; Ho, 1998)

The disagreement measure is the ratio of the number of examples labelled by f_i and f_k

differently (i.e. one is correct, and the other is incorrect) to the total number of examples. It refers to the probability that two classifiers will disagree.

$$Dis_{i,k} = \frac{N^{01} + N^{10}}{N^{11} + N^{10} + N^{01} + N^{00}}. \tag{2.9}$$

(4) Double-fault (DF) (Giacinto and Roli, 2000)

The double-fault measure is the proportion of examples that are labelled incorrectly by both classifiers,

$$DF_{i,k} = \frac{N^{00}}{N^{11} + N^{10} + N^{01} + N^{00}}. \tag{2.10}$$

The above four statistics belong to pairwise measures that express the (dis)similarity between any two classifiers. The following gives six non-pairwise measures that take the ensemble's decision into consideration. Let $l\left(x_j\right)$ denote the number of classifiers that label x_j correctly (i.e. $l\left(x_j\right) = \sum_{i=1}^{L} O_{j,i}$).

(5) Entropy (Ent) (Cunningham and Carney, 2000)

The entropy measure assumes that the highest diversity happens when half of the classifiers are correct and the remaining ones are incorrect. If the oracle outputs are all 1's or 0's, there is no diversity among the classifiers. It is defined as

$$Ent = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{\left(L - \lceil L/2 \rceil\right)} \min\left\{l\left(x_j\right), L - l\left(x_j\right)\right\}, \tag{2.11}$$

where $\lceil . \rceil$ is the ceiling operator. Ent varies between 0 and 1, where 0 indicates all classifiers perform the same, and 1 indicates the highest diversity.

(6) Kohavi-Wolpert variance (KW) (Kohavi and Wolpert, 1996)

The Kohavi-Wolpert variance measures the variability of the predicted class label given an input x for a specific classifier

$$variance_x = \frac{1}{2}\left(1 - \sum_{i=1}^{c} P\left(y = \omega_i|x\right)^2\right) \tag{2.12}$$

and averages over the whole data set. In the case of oracle outputs, the variability is estimated among the set of classifiers $f_1, f_2, \ldots, f_L$ by

$$KW = \frac{1}{NL^2}\sum_{j=1}^{N} l\left(x_j\right)\left(L - l\left(x_j\right)\right). \tag{2.13}$$

It can be proved that KW differs from the averaged disagreement measure Dis by a constant factor (Kuncheva and Whitaker, 2003),

$$KW = \frac{L-1}{2L} Dis. \tag{2.14}$$

(7) Interrator agreement (κ) (Fleiss, 1981)

It is a measure of classification reliability, indicating the level of agreement while corrected by chance. Let $\bar{p}$ denote the average individual classification accuracy

$$\bar{p} = \frac{1}{NL}\sum_{j=1}^{N} l\left(x_j\right), \tag{2.15}$$

then

$$\kappa = 1 - \frac{\frac{1}{L}\sum_{j=1}^{N} l\left(x_j\right)\left(L - l\left(x_j\right)\right)}{N\left(L-1\right)\bar{p}\left(1-\bar{p}\right)}. \tag{2.16}$$

κ is related to KW and Dis as follows

$$\kappa = 1 - \frac{L}{\left(L-1\right)\bar{p}\left(1-\bar{p}\right)} KW = 1 - \frac{1}{2\bar{p}\left(1-\bar{p}\right)} Dis. \tag{2.17}$$

It should be noted that there is a pairwise κ statistic used in (Margineantu and Dietterich, 1997; Dietterich, 2000a). It is the agreement between two classifiers corrected by chance. However, these two have no direct link.

(8) Difficulty measure (θ) (Hansen and Salamon, 1990)

A discrete random variable X' is defined as the proportion of classifiers in F that classify an input x correctly: $X' = \left\{\frac{0}{L}, \frac{1}{L}, \ldots, 1\right\}$. It describes a pattern of classification difficulty for all classifiers. The diversity measure θ depicts the distribution of difficulty over the data set Z, captured by the variance of X': $\theta = Var\left(X'\right)$. The higher the value of θ, the lower the ensemble diversity.

(9) Generalized diversity (GD) (Partridge and Krzanowski, 1997)

Partridge and Krzanowski argued that maximum diversity occurs when failure of one of the L classifiers is accompanied by correct labeling by the other classifier; minimum diversity occurs when one's failure is always accompanied by failure of the other. If we define p_i as the probability that i out of L classifiers misclassify an input x simultaneously and $p\left(i\right)$ as the probability that i randomly chosen classifiers will misclassify x,

$$p\left(1\right) = \sum_{i=1}^{L} \frac{i}{L} p_i \text{ and } p\left(2\right) = \sum_{i=1}^{L} \frac{i\left(i-1\right)}{L\left(L-1\right)} p_i. \tag{2.18}$$

In the case with maximum diversity, there is $p\left(2\right) = 0$; in the case with minimum diversity, $p\left(2\right) = p\left(1\right)$ holds. Based on the assumptions, the generalization diversity is defined as

$$GD = 1 - \frac{p\left(2\right)}{p\left(1\right)}. \tag{2.19}$$

GD varies between 0 and 1.

(10) Coincident failure diversity (CFD) (Partridge and Krzanowski, 1997)

The coincident failure diversity is a modification of GD. It is defined as

$$CFD = \begin{cases} 0, & p_0 = 1.0 \\ \frac{1}{1-p_0} \sum_{i=1}^{L} \frac{L-i}{L-1} p_i, & p_0 < 1.0 \end{cases} \tag{2.20}$$

It varies between 0 and 1. When all classifiers always produce correct labels or they are correct or wrong simultaneously (minimum diversity), CFD will be 0. When all the misclassifications happen to exactly one classifier (maximum diversity), 1 will be achieved.

Table 2.5 summarizes the ten measures, including the indication of their changing directions in relation to diversity, the category they belong to (pairwise or not) and the abbreviations used in this thesis.

Table 2.5: Summary of the 10 diversity measures. '↓/↑' indicates greater diversity if the measure gets lower/higher.

Name	Abbr.	Pairwise	Direction
Q-statistic	Q	Yes	↓
Correlation coefficient	ρ	Yes	↓
Disagreement measure	Dis	Yes	↑
Double-fault	DF	Yes	↓
Entropy	Ent	No	↑
Kohavi-Wolpert variance	KW	No	↑
Interrator agreement	κ	No	↓
Difficulty measure	θ	No	↓
Generalized diversity	GD	No	↑
Coincident failure diversity	CFD	No	↑

Kuncheva and Whitaker studied the relationship between these diversity measures (Kuncheva and Whitaker, 2003). Strong positive correlations were observed. They found no single best measure. Similar results were also obtained in other papers (Shipp and Kuncheva, 2002a; Chen, 2008). Particularly, Q-statistic was recommended by the authors for its simplicity and understandability (Kuncheva and Whitaker, 2003). Besides, Q-statistic was shown to be the only measure, whose maximum, minimum and independence value do not depend on the individual accuracy for two classifiers with the same individual accuracy (Kuncheva and Whitaker, 2001), and have little empirical relationship to the averaged individual accuracy (Kuncheva and Whitaker, 2003). It could be an advantage for studying class imbalance problems, considering that a diversity measure less correlated to individual accuracy is expected to be less sensitive to class imbalance distributions. Otherwise, it may result in inaccurate analysis.

2.3.2 Diversity and Generalization

In the regression context, diversity is clearly related to the covariance term of the ensemble error. It explains the role of the differences between individual learners in generalization performance based on two decompositions of the quadratic loss error function (Brown et al., 2005) – the bias-variance-covariance decomposition (Ueda and Nakano, 1996) and the ambiguity decomposition (Krogh and Vedelsby, 1995). They will be introduced in section 2.4 as the theoretical grounding of negative correlation learning.

Unlike regression ensembles, there is no clear analogue of such decompositions in the classification context. The misclassification rate is an appropriate and the most common performance evaluation criterion in a classification task, as known as the *zero-one error function*. A prediction for a given example can either be correct (0 error penalty) or wrong (1 error penalty). The error rate is estimated based on the count of committed misclassifications. Following the notations in the previous section, we let y_j be the true class of any example x_j, and $y_f(x_j)$ be the class label produced by the classifier f. y_j and $y_f(x_j)$ belong to the output space $\{\omega_1, \omega_2, \ldots, \omega_c\}$. Then the 0-1 error of f on x_j is defined as (Domingos and Pazzani, 1997)

$$L(f, x_j) = \begin{cases} 0 & \text{if } y_f(x_j) = y_j \\ 1 & \text{if } otherwise. \end{cases} \tag{2.21}$$

In general, if x_j is associated with a vector of class probabilities, let $P(\omega_i|x_j)$ $(i = 1, \ldots, c)$

denote the true values of the posterior probabilities for the c classes from the target function y, and $P_f(\omega_i|x_j)$ denote the probability of f assigning class ω_i to x_j. The 0-1 error can be more generally defined as (Kohavi and Wolpert, 1996; Domingos and Pazzani, 1997)

$$L(f, x_j) = 1 - \sum_{\omega_i} P_f(\omega_i|x_j)\, P(\omega_i|x_j) \tag{2.22}$$

based on the proposition that "*labeling x_j as ω_i and x_j belonging to ω_i are conditionally independent given the target function y and input x_j*" (Kohavi and Wolpert, 1996). The second term is referred to as the accuracy of f on x_j. This definition reduces to Eq. 2.21 when one class has probability 1.

There is no unique bias-variance decomposition for this type of loss (Kuncheva, 2004), and thus no simple and clear accuracy-diversity breakdown has been achieved for classification ensembles. To study the role of diversity in generalization theoretically, existing studies have given separate focuses on classifiers producing class probabilities (real-valued outputs) and classifiers producing discrete class labels with their own assumptions and achieved highly restricted theoretical results. Work on this topic is still progressing.

Real-Valued Outputs with a Linear Combination

When base classifiers can output real-valued numbers that approximate posterior probabilities of each class, Tumer and Ghosh reformulated the *added error* rate above the *Bayes error* of an ensemble with a simple averaging combination method (Tumer and Ghosh, 1996a, 1999). Bayes error is irreducible under the Bayes optimum decision. The added error is the quantity made by a classifier beyond the Bayes error. The expected added error of averaging L unbiased classifiers is

$$E_{add}^{ave} = E_{add}\left(\frac{1 + \delta(L-1)}{L}\right), \tag{2.23}$$

where E_{add} is the expected added error of individual classifiers. δ is the average of the class-specific correlations among classifiers. When the individuals are identical, δ is equal to 1. The ensemble error is just the individual error in this case. When the individuals are statistically independent with $\delta = 0$, the ensemble error is reduced by a factor L. When the individuals are negatively correlated with a negative δ, the ensemble error is lower than the average individual error. However, the derivation of Eq. 2.23 is based on some strong assumptions. Some extension was done by Fumera and Roli for ensembles with a weighted averaging combination (Fumera and Roli, 2003, 2005).

Discrete Outputs with a Majority Voting Combination

When classifiers can only output discrete labels, such as k-nearest neighbour, the relationship between diversity and the ensemble error becomes more vague. Kuncheva et al. built mathematical links for diversity measured by Q-statistic to the majority vote accuracy in some special cases (Kuncheva et al., 2003). They proved that increasing diversity is not always beneficial to overall accuracy. Two extreme patterns of ensembles with different voting combinations were found. They were claimed to have the possible characteristics of the "best" and "worst" combinations of individual classifiers when they hold the same accuracy. In the "best" pattern, reducing Q-statistic (i.e. larger diversity) will improve overall accuracy; in the "worst" pattern, it will result in worse accuracy. More details of each pattern will be given in chapter 3, in which we exploit the idea of patterns to study the relationship between diversity and single-class performance in class imbalance learning.

Chung et al. established upper and lower bounds for the majority vote accuracy in terms of average individual accuracy and the diversity measure of entropy (Chung et al., 2007). They showed that the ensemble accuracy is an increasing function with respect to the entropy measure when the individual accuracy is large enough.

So far, we have introduced the concept of diversity in the classification context, including various definitions and its theoretical role in overall performance of ensembles under different assumptions. The lack of a unified definition and theoretical framework of diversity for classification problems makes it less useful for designing a good ensemble algorithm.

2.4 Negative Correlation Learning Algorithms

Negative correlation learning (NCL) is a successful neural network (NN) ensemble technique that manages ensemble diversity among the NN members explicitly in addition to the individual accuracy during the learning procedure. The principle underlying NCL can be justified by decomposing the generalization error of regression ensembles (Brown et al., 2005). In addition to the error bias and variance of each individual learner, the mean squared error (MSE) of ensemble critically depends on the covariance between the individuals (Ueda and Nakano, 1996), corresponding to diversity. NCL captures this effect with the aim of balancing covariance against bias and variance. Certain NCL algorithms have been proposed to *negatively correlate* the errors made by each other and encourage the interaction of individuals.

Liu and Yao's cooperative ensemble learning system (CELS) (Liu and Yao, 1999b) is a representative algorithm with the standard NCL paradigm. Unlike Bagging (Breiman, 1996) and Boosting (Freund and Schapire, 1996), the idea of CELS is to encourage the individual networks to learn different aspects of a given data set cooperatively and simultaneously through the gradient descent procedure. It introduces a penalty term as a regularization factor into the error function of each network. It contains the information of error correlation in the ensemble by exploiting the well-known ambiguity decomposition (Krogh and Vedelsby, 1995). Adding this penalty term fills the missing gradient component in relation to variance and covariance terms in the ensemble MSE (Brown et al., 2005). It achieved empirical success in both regression and classification problems. The algorithm is described in Table 2.6.

The error function e_i of network i in step 2 is formed by two terms. The first term is its empirical training error. The second term p_i is the correlation penalty function.

Table 2.6: The CELS algorithm (standard NCL paradigm) (Liu and Yao, 1999b).

```
Given: a training set $Z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$; the number of neural networks $L$; the learning rate $\eta$ for backpropagation (BP); the penalty coefficient $\lambda$.
Initialize $L$ neural networks $f_1, \ldots, f_L$.

For each training example $n = 1, 2, \ldots, N$:
Step 1. Calculate $\bar{f}(x_n) = \frac{1}{L}\sum_i f_i(x_n)$.
Step 2. For each network $i = 1, 2, \ldots, L$:
        Update each weight $w$ in network $i$ using learning rate $\eta$ by computing
        (1) Individual error function $e_i(x_n) = \frac{1}{2}(f_i(x_n) - y_n)^2 + \lambda p_i(x_n)$,
        where the penalty term $p_i(x_n) = \left(f_i(x_n) - \bar{f}(x_n)\right)\sum_{j \neq i}\left(f_j(x_n) - \bar{f}(x_n)\right)$,
        (2) $\frac{\partial e_i(x_n)}{\partial f_i(x_n)} = (f_i(x_n) - y_n) + \lambda\sum_{j \neq i}\left(f_j(x_n) - \bar{f}(x_n)\right)$,
        (3) $\Delta w = -\eta\left[(f_i(x_n) - y_n) - \lambda\left(f_i(x_n) - \bar{f}(x_n)\right)\right]\frac{\partial f_i(x_n)}{\partial w}$.
Repeat the above for desired iterations.

Output of the NCL ensemble for any new example $x$: $f(x) = \frac{1}{L}\sum_i f_i(x)$.
```

Minimizing p_i aims to negatively correlate network i's error with the errors of the other members. Parameter λ controls the trade-off between the two terms ranging in $[0, 1]$.

CELS was found computationally expensive when dealing with large data sets due to the pattern-by-pattern weight-updating strategy. To speed up the learning process, another NCL algorithm, negative correlation learning via correlation-corrected data (NCCD), was proposed by Chan and Kasabov (Chan and Kasabov, 2005b,a). The idea is to incorporate error correlation information into training data instead of every network's error function. The training data with adjusted outputs is called Correlation-Corrected data (C-C data). Each network is provided with its own C-C data that are updated after every certain training epochs. It reduces the updating times of information exchange and allows parallel model implementation. The training strategy is summarized in Table 2.7.

The equation in step 2 is derived by making $\frac{\partial \bar{e}}{\partial f_i} = 0$ to generate C-C data, where $\bar{e}$ is the ensemble error. The penalty coefficient λ varies in $[0, 1)$. NCCD was shown to have comparable performance to CELS and much less computation overhead.

Besides CELS and NCCD, there are other variations with the NCL paradigm. Islam et al. proposed a constructive method for training cooperative neural network ensem-

Table 2.7: The NCCD algorithm (Chan and Kasabov, 2005a).

```
Given: a training set $Z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$; the number of neural
networks $L$; the penalty coefficient $\lambda$; the updating interval of C-C data $g$.
Initialize $L$ neural networks $f_1, \ldots, f_L$ and train them individually for $g$ epochs.

For network $i$:
Step 1. Calculate $\bar{f}(x_n) = \frac{1}{L}\sum_i f_i(x_n)$.
Step 2. Create C-C data by updating the targets of training data $y$ to $y'$:
        $y' = \frac{y-\lambda\bar{f}}{1-\lambda}$.
Step 3. Use C-C data $\{(x_i, y')\}$ to train network $i$ for $g$ epochs.
Repeat the above until every network has trained for desired iterations.

Output of the NCL ensemble for any new example $x$: $f(x) = \frac{1}{L}\sum_i f_i(x)$.
```

bles (Islam et al., 2003). It combines the architecture design of neural networks with NCL training, in which the number of networks and the number of hidden nodes in each individual are determined automatically. Further, they applied this strategy to Bagging and Boosting (Islam et al., 2008). Dam et al. applied NCL to learning classifier systems for compacting rules, where NCL was shown to improve ensemble generalization (Dam et al., 2007). Chen and Yao found that NCL could suffer from the problem of overfitting noise. They proposed regularized negative correlation learning (RNCL) algorithm that introduces an additional regularization term into the error function of the whole ensemble, along with a Bayesian interpretation (Chen and Yao, 2009).

The theoretical foundation of the "error correlation" term used in NCL comes from two error decompositions in the regression context as we have mentioned before: bias-variance-covariance decomposition (Ueda and Nakano, 1996) and ambiguity decomposition (Krogh and Vedelsby, 1995). The ambiguity decomposition states that the quadratic error of the ensemble is less than or equal to the average quadratic error of individual learners:

$$\left(\bar{f} - y\right)^2 = \sum_i \alpha_i \left(f_i - y\right)^2 - \sum_i \alpha_i \left(f_i - \bar{f}\right)^2, \tag{2.24}$$

where $\bar{f}$ is the weighted average of the individuals $\bar{f} = \sum_i \alpha_i f_i$. The second term on the right-hand side of the equation is referred to as the *ambiguity* term. It measures the disagreement or prediction variability within the ensemble and guarantees that the ensemble has a lower error than the average individual error.

The role of ensemble diversity in generalization can be explained by the bias-variance-covariance decomposition (Ueda and Nakano, 1996). For a uniformly weighted ensemble $\bar{f}$ with L individual members, its mean squared error can be decomposed into

$$E\left\{\left(\bar{f}-y\right)^2\right\} = \overline{bias} + \frac{1}{L}\overline{var} + \left(1-\frac{1}{L}\right)\overline{covar}, \tag{2.25}$$

where

$\overline{bias} = \frac{1}{L}\sum_i \left(E\left\{f_i\right\} - y\right)$, the averaged bias of the individuals;

$\overline{var} = \frac{1}{L}\sum_i E\left\{\left(f_i - E\left\{f_i\right\}\right)^2\right\}$, the averaged variance of the individuals;

$\overline{covar} = \frac{1}{L(L-1)}\sum_i \sum_{j\neq i} E\left\{\left(f_i - E\left\{f_i\right\}\right)\left(f_j - E\left\{f_j\right\}\right)\right\}$, the averaged covariance between the individuals.

It tells us that in addition to the bias and variance from the individuals, the generalization error also depends on how they correlate to each other. Inspired by the decompositions, NCL divides the ensemble ambiguity and assigns each portion to every ensemble member. This explains where the penalty term comes from. Brown et al. gave the exact link between these two decompositions (Brown et al., 2005; Brown, 2004), showing that we could not simply maximize the ambiguity without affecting the individual error term. It implies a trade-off. The link can be found in the Appendix. A section.

Although NCL algorithms are theoretically supported in the regression context and have been applied to classification problems, they are still theoretically unsound in the classification context, because of the difficulty of achieving a clear accuracy-diversity breakdown of the classification ensemble error, as we have reviewed in section 2.3.2. In addition to an unclear theoretical framework for classification, existing NCL algorithms confront some other known drawbacks. They are restricted to the use of neural networks as the base learner based on gradient descent. The capability of processing real-valued outputs is required. Other base learners, such as decision trees and k-NN, are obviously not suitable to this training paradigm. Poor flexibility and low efficiency hinder them from being more widely used. With the aim of overcoming these problems, chapter 4 proposes a new NCL algorithm for classification ensembles, AdaBoost.NC.

2.5 Chapter Summary

This chapter reviewed the existing studies concerning class imbalance learning and ensemble learning. We first reviewed the current research of learning from class imbalance problems, including data-level, algorithm-level and ensemble-based methods and their reported issues, from which we are motivated to search for a better solution and study ensemble diversity for class imbalance problems. Meanwhile, we noticed that the majority of class imbalance learning studies are restricted to the discussions of two-class problems (i.e. data with one minority class and one majority class). However, many practical applications that suffer from the classification difficulty of imbalanced distributions contain more than two classes. We therefore reviewed the up-to-date progress in multi-class imbalance learning. Among limited solutions for multi-class imbalance problems, most efforts have been focused on class decomposition schemes. It is desirable to develop a direct and effective approach. We highlighted the challenges on this topic that will be looked into in the later chapter as a necessary part of class imbalance learning. How to evaluate class imbalance learning algorithms in both two-class and multi-class scenarios were included in this chapter.

In order to make best use of ensemble diversity in class imbalance learning, we need a better understanding of this concept in the classification context. Thus, following the introduction of class imbalance learning, we gave the definitions of diversity in classification ensembles and explained its role in generalization performance, which will be used to study the effect of diversity on the classification of class imbalance problems in chapter 3. Finally, we described the idea of negative correlation learning as an important ensemble technique that emphasizes diversity explicitly during learning, and introduced existing

NCL algorithms in the literature. Under this topic, the identified research issues motivate us to develop a better NCL algorithm for classification ensembles in chapter 4. This new learning algorithm will be explored and exploited to tackle both two-class and multi-class imbalance problems in chapters 5 and 6.

CHAPTER 3

DIVERSITY ANALYSIS FOR CLASS IMBALANCE LEARNING

In the previous chapter, we introduced a number of ensemble approaches to tackling class imbalance problems. They adapt individual classifiers to emphasize the minority class and utilize the difference among individuals to reduce the risk of overfitting and stabilize the performance. In this chapter, we would like to understand the effect of ensembles in depth, i.e. the contribution of diversity, in the context of class imbalance learning. Section 3.1 gives a generic framework for defining class imbalance learning in terms of decision theory to help us understand the learning task and objective precisely. Section 3.2 discusses the relationship between diversity and overall accuracy of ensembles in classification patterns by providing theoretical links. It is extended to single-class performance in section 3.3. Section 3.4 verifies the theoretical results of single-class performance and studies the impact of diversity on overall performance through extensive experimental work in class imbalance scenarios, followed by the chapter summary in section 3.5. The research questions in section 1.3.1 are answered here.

3.1 Introduction

For a class imbalance problem, the learning objective is generally described as “obtaining a classifier that will provide high accuracy for the minority classes without jeopardizing the accuracy of the majority classes (He and Garcia, 2009)". It emphasizes two points. First, minority classes deserve more attention. Misclassifying a minority class example has a higher cost than misclassifying an example belonging to the majority class. Second, good overall performance should be maintained while the focus is on the minority class. Based on this understanding and Bayesian decision theory, we formulate the class imbalance problem as follows.

Given an imbalanced data set with c possible class labels, denoted by $Y = \{\omega_1, \omega_2, \ldots, \omega_c\}$ $(c \geq 2)$, we suppose that the first k classes $\{\omega_1, \ldots, \omega_k\}$ $(1 \leq k < c)$ of examples are heavily under-represented in comparison with the other classes $\{\omega_{k+1}, \ldots, \omega_c\}$. In most real-world applications, this information is usually provided by domain experts as prior knowledge. However, we may not know which minority classes suffer from severe classification difficulties and what the exact misclassification costs are. The task is to assign one of the c labels to any input example x. For clear presentation, we establish some notations here.

- $p(\omega_i|x)$: the posterior probability of each class ω_i produced by the ideal classifier (the true class distribution).

- $C(\omega_i, \omega_j)$: the cost of predicting class ω_i when the true class of x is ω_j.

- $L(x, \omega_i)$: the loss of assigning label ω_i to example x.

- $R(x, \omega_i)$: the probability of predicting class ω_i when the true class of x is ω_i. When the context is clear, R_i is used.

- $P(x, \omega_i)$: the probability of x having the true class ω_i when it is predicted as ω_i. When the context is clear, P_i is used.

Based on decision theory, the optimal prediction for example x should be class ω_i that minimizes (Elkan, 2001)

$$L(x, \omega_i) = \sum_j p(\omega_j|x) \, C(\omega_i, \omega_j) \,. \tag{3.1}$$

In class imbalance learning, however, the costs are not often provided explicitly. We need to reformulate $C(\omega_i, \omega_j)$. $C(\omega_i, \omega_j)$ is the cost of classifying an example x with real label ω_j to class ω_i. When it happens, R_j and P_i are supposed to get lower. On one hand, input x belonging to class ω_j is misclassified, which implies smaller R_j. On the other hand, a wrong-labelled example is added into the set with the predicted label of class ω_i, which indicates smaller P_i. Therefore, we re-express $C(\omega_i, \omega_j)$ as a function of $\left(\frac{1}{R_j}, \frac{1}{P_i}\right)$,

$$C(\omega_i, \omega_j) = g\left(\frac{1}{R_j}, \frac{1}{P_i}\right), \tag{3.2}$$

where g is a monotone decreasing function of R_j and P_i. Correspondingly, the loss function becomes

$$L(x, \omega_i) = \sum_j p(\omega_j|x)\, g\left(\frac{1}{R_j}, \frac{1}{P_i}\right). \tag{3.3}$$

In order to minimize the misclassification cost, the optimal decision will be

$$\omega_{opt} = \arg\underset{\omega_i}{min} \sum_j p(\omega_j|x)\, g\left(\frac{1}{R_j}, \frac{1}{P_i}\right). \tag{3.4}$$

In class imbalance learning, therefore, the objective can be interpreted as "*obtaining a classifier that maximizes R_j's and P_i's*", when the cost is not available.

With this mathematical framework, let's see how to make a decision for an imbalanced problem. If ω_j is one of the minority classes ($j \leq k$), it is reasonable to assume that it has a small R_j but a relatively good P_j, due to its poor representation in the data set. Correspondingly, if ω_i is one of the majority classes ($i > k$), it tends to have a relatively small P_i and a high R_i, because most examples in this class are easier to identify. Referring to the function g, $C(\omega_i, \omega_j)$ should be larger than $C(\omega_j, \omega_i)$, which tallies with the essential understanding of class imbalance learning – again, misclassifying a minority class example has a higher cost than misclassifying a majority class example. As a result, the loss of predicting class ω_i will become high. In fact, the misclassification cost in class imbalance learning is associated with the relative classification difficulty of class pairs in this problem formulation framework. For some easy situations, the small size of minority class may not be a problem. When the difficulty exists, the loss of producing the majority class label for any input belonging to the minority class will be raised up by the cost function.

R_i and P_i correspond to recall and precision of class ω_i, which are two important single-class performance measures for classifier evaluation in class imbalance learning, as we have introduced in section 2.1.3. Therefore, *the ideal classifier should be the best at balancing the recalls and precisions over all classes.*

With a better understanding of class imbalance learning, we now divert our attention to ensemble solutions for class imbalance problems and study if and how ensemble diversity helps to achieve the learning objective. Most existing ensemble algorithms adjust the learning bias of individual classifiers towards the majority class by manipulating training data differently from the data level (Li, 2007; Liu et al., 2009; Chawla et al., 2003) or by applying different costs from the algorithm level (Joshi et al., 2001) (Chawla and Sylvester, 2007; Guo and Viktor, 2004; Fan et al., 1999). Combining multiple classifiers can reduce the probability of overfitting (Perrone and Cooper, 1993). The interaction among them is able to make up each other's weaknesses (Kotsiantis et al., 2006; Kotsiantis and Pintelas, 2003; Estabrooks et al., 2004; Seiffert et al., 2008a,b). Thus, an ensemble should have better performance than any individual on average by making use of this interaction (Estabrooks et al., 2004; Kotsiantis and Pintelas, 2003; Chawla and Sylvester, 2007). It is reflected in the covariance of the ensemble members, i.e. ensemble diversity in ensemble learning. However, no study has actually investigated its effects in classifying imbalanced data sets so far.

To date, existing studies have been attempting to relate diversity of classification ensembles to overall accuracy by giving separate focuses on classifiers producing class probabilities (real-valued outputs) (Tumer and Ghosh, 1996a, 1999, 1996b) and classifiers producing discrete class labels (Kuncheva et al., 2003) under certain assumptions. In class imbalance cases, however, the overall accuracy is not appropriate and less meaningful (Kotsiantis et al., 2006). Single-class performance measures are adopted to evaluate how a classifier performs in the specific class we are concerned with. Recall, precision and F-measure (Rijsbergen, 1979) are most widely used in the class imbalance learning literature.

Accordingly, this chapter will provide answers to the following questions: 1) what is the relationship between ensemble diversity and the performance measures used in class imbalance learning? 2) Is introducing diversity beneficial to the minority/majority class in the presence of imbalanced data?

To answer the first question, section 3.2 discusses when and why diversity measured by Q-statistic (Yule, 1900) causes better overall accuracy based on several classification patterns of ensembles by extending Kuncheva et al.'s study (Kuncheva et al., 2003). We explain why diversity is not always beneficial to overall accuracy. Two arguments are proposed in patterns accordingly for the minority and majority classes respectively of a class imbalance problem. The pattern analysis is then utilized to relate Q-statistic to single-class performance measures, including recall, precision and F-measure in section 3.3. We show mathematically how these single-class measures behave as diversity varies. Six possible situations with different changing behaviours are obtained and analyzed. The relationship between diversity and overall performance measured by AUC and G-mean is discussed empirically in section 3.4, where diversity is shown to be beneficial.

To answer the second question of how diversity affects single-class performance of each type of classes in the presence of imbalanced data, comprehensive experiments are carried out on artificial and real-world data sets with highly skewed class distributions in section 3.4. We find strong correlations between diversity measured by the pairwise measure Q-statistic and the non-pairwise measure GD and discussed performance measures. Diversity shows a positive impact on the minority class in terms of recall and F-measure in general, which is achieved by making the ensemble produce broader and less overfitting classification boundaries for the minority class. This chapter focuses on binary classification problems.

In the following relationship study of this chapter, Q-statistic is chosen as the main diversity measure from various definitions for the following reasons: 1) strong positive correlations among existing diversity measures have been found mathematically and empirically. They share very high similarity. Q-statistic was particularly recommended for its simplicity of calculation and understandability (Kuncheva and Whitaker, 2003). 2) Q-statistic has shown little relationship to the individual accuracy (Kuncheva and Whitaker, 2001, 2003). It could be an advantage for studying diversity in the class imbalance learning context, because a measure depending on individual accuracy is likely to be sensitive to imbalanced distributions and cause misleading conclusions. 3) All the mathematical equations in the pattern analysis later on are derived in terms of Q-statistic. It would be better to use it throughout the chapter for consistency. To confirm our correlation results based on Q-statistic, we further examine a non-pairwise diversity measure "generalized diversity (GD)" in the experimental analysis. The definitions of Q-statistic and GD can be found in section 2.3.1.

3.2 Relationship Between Diversity and Overall Accuracy

In this section, we explain the functional relationship between ensemble diversity and overall accuracy in several patterns by extending Kuncheva et al.'s study (Kuncheva et al., 2003), including two extreme patterns and a general pattern. A classification pattern refers to the voting combinations of the individual classifiers that an ensemble can have. The accuracy is given by the majority voting method of combining classifier decisions. Two extreme patterns present different effects of diversity, which shows that diversity is not always beneficial to the generalization performance. The reason is explained in a general pattern. Based on the features of the patterns, we relate them to the classification of each type of classes of a class imbalance problem, and two arguments are proposed accordingly.

3.2.1 Definitions and Notations

We give some notations first for the following discussions. Suppose a data set $Z = \{z_1, \ldots, z_N\}$ ($z_j = (x_j, y_j)$ and cardinality $|Z| = N$) with two possible labels, positive class "+1" and negative class "-1". By convention, we treat the positive class "+1" as the minority class that forms the minority-class subset Z_{min}. The remaining examples with label "-1" form the majority-class subset Z_{maj}. Single-class measures discussed in this chapter, recall, precision and F-measure, are denoted by 'R', 'P' and 'F' respectively. Q-statistic is denoted by Q. We use a "min" subscript to denote a single-class measure evaluated on the minority class, and a "maj" subscript to denote "majority". For example, R_{min} and R_{maj} indicate the recall measure of the minority class and majority class respectively. For an ensemble $\bar{f}$ composed of L individual classifiers $\{f_1, f_2, \ldots, f_L\}$, the ensemble size L is restricted to an odd number for the convenience of calculation later. Let $l = \lfloor L/2 \rfloor$. We also define,

- θ: the imbalance rate of the given data set, defined as $|Z_{min}|/|Z|$ ($\theta < 0.5$). θ is equal to 0.5 when data is balanced.

- P_{ovr}: the overall accuracy of the ensemble with the combination method of majority vote from individual classifiers. Every classifier produces a class label and they equally contribute to the final output.

- p: the overall accuracy of the individual classifier. A classifier has probability p of giving the correct label to any input. In Kuncheva et al.'s two extreme patterns, L ensemble members are assumed to have the same individual accuracy (Kuncheva et al., 2003). All the patterns discussed in this chapter are based on this assumption.

- p_{ab}: the occurrence probability of the respective combination of correct and wrong outputs for any pair of classifier f_i and f_k. 'a' and 'b' belong to {1 (correct label), 0 (incorrect label)}. For example, p_{01} stands for the probability of f_i being incorrect and f_k being correct.

3.2.2 Q-statistic and Overall Accuracy in Patterns

The relationship between Q-statistic and overall accuracy can be explained in two extreme situations and is summarized into a general pattern. The two extreme situations refer to a best pattern and a worst pattern (Kuncheva et al., 2003), which will be introduced first. Explicit mathematical links between Q and P_{ovr} exist, and Q exhibits different impacts on P_{ovr} in the patterns. Then, they will be analyzed in a more general context, in which we provide a deeper understanding of why accuracy behaves differently with respect to Q in different patterns. A good pattern and a bad pattern are defined accordingly.

Two Extreme Patterns

Kuncheva et al. defined and analyzed two probability distributions over the possible combinations of L votes from the ensemble members, referred to as "pattern of success" and "pattern of failure" (Kuncheva et al., 2003). The two patterns were claimed to have the possible characteristics of the "best" and "worst" combinations of L classifiers when they hold the same accuracy p.

(1) Pattern of success (best pattern): In this pattern, no correct votes are "wasted", which means there are exactly $l+1$ classifiers giving correct answers to every correctly predicted example, or all of the classifiers give wrong answers if the combined result is wrong. Any extra correct vote will be a waste. To give a formal definition (Kuncheva et al., 2003),

- The probability of any combination with $l+1$ correct and l incorrect votes is α_0.
- The probability of all L votes being incorrect is β_0.
- The probability of all other combinations is 0.

Theorem 3.1. *Under the best pattern, the expression of P_{ovr} with respect to Q is (Kuncheva et al., 2003)*

$$P_{ovr} = \frac{L}{(l+1)} \frac{(1-Q)}{(2-Q)}. \tag{3.5}$$

P_{ovr} is a monotone decreasing function of the pairwise dependence Q. For any $p > 2/3$, the value of Q will be -1.

(2) Pattern of failure (worse pattern): In this pattern, correct votes are "wasted" to the maximum extent. All the classifiers give correct answers if the combined result is correct. Otherwise, there are exactly l correct votes, which produce the wrong class label. It is defined as (Kuncheva et al., 2003),

- The probability of all L votes being correct is α_l.
- The probability of any combination with l correct and $l+1$ incorrect votes is β_l.
- The probability of all other combinations is zero.

Theorem 3.2. *Under the worst pattern, the expression of P_{ovr} with respect to Q is (Kuncheva et al., 2003)*

$$P_{ovr} = \frac{L}{(l-1)}\frac{1}{(2-Q)} - \frac{l}{(L-l)}, \tag{3.6}$$

P_{ovr} is a monotone increasing function of Q. For any $p > 0.5$, the value of Q is positive.

The patterns show that diversity does not always produce a positive effect on ensembles, which depends on how many correct votes from the individual classifiers are wasted. When diversity is "good", such as the best pattern, diversity causes a performance improvement. Otherwise, there is some "bad" diversity that harms the performance. The worst pattern is one of such cases. It is a useful analysis since it finds when diversity can be beneficial or harmful, and provides the possible factor (i.e. the number of wasted correct votes) that is somehow related to its effect. However, the original paper did not explain why the number of wasted correct votes with a corresponding voting pattern of the ensemble is possible to discriminate ensembles with different impacts of diversity.

Let's take a closer look of these patterns and corresponding functions. In fact, the number of wasted correct votes is related to which voting combination low diversity resides in. α and β decide the ensemble accuracy. In the best pattern, low diversity exists in the voting combination on misclassified examples, where every classifier agrees on the wrong label. Increasing diversity makes the wrong voting combination with low disagreement degree less likely to happen and enforces the correct voting combination with high disagreement degree (i.e. higher α and lower β). Therefore, the overall accuracy gets improved. On the contrary, low diversity results from the correct voting combination in the worst pattern. Encouraging diversity increases the probability of occurrence of the wrong voting combination β, thus worse accuracy obtained. This may provide us with some clues of the link between the definition of the patterns and the impact of diversity.

General Pattern

To further understand the patterns in a general sense, we define a general pattern by allowing more than two possible voting combinations of an ensemble. Although it is hard to derive a neat expression for Q-statistic and accuracy under this pattern, it shows how they are related through types of voting combinations and their possibilities. It is defined as follows,

- Given a voting combination that provides the correct class label for an input, the number of correct votes is $l+1+i$ with probability α_i $(i = 0, 1, \ldots l)$.
- Given a voting combination that provides the incorrect class label for an input, the number of correct votes is j with probability β_j $(j = 0, 1, \ldots l)$.

α_i and β_j determine the voting distribution of the ensemble. Their subscripts i and j indicate the number of wasted correct votes in the corresponding voting combination. We apply the same inferring method as in the extreme patterns (Kuncheva et al., 2003). P_{ovr} is the sum of the probability of each correct voting combination. There are $\binom{L}{l+1+i}$ ways of having $(l+1+i)$ correct out of L classifiers, each with probability α_i. The majority vote accuracy becomes

$$P_{ovr} = \sum_{i=0}^{l} \binom{L}{l+1+i} \alpha_i \tag{3.7}$$

with condition $\sum_{i=0}^{l}\binom{L}{l+1+i}\alpha_i+\sum_{j=0}^{l}\binom{L}{j}\beta_j=1$.

Table 3.1: Pairwise table of the relationship between any two classifiers for the general pattern.

	f_k	
f_i	1	0
1	$p_{11}=\sum_{i=0}^{l}\binom{L-2}{l-1+i}\alpha_i+$ $\sum_{j=2}^{l}\binom{L-2}{j-2}\beta_j$	$p_{10}=\sum_{i=0}^{l-1}\binom{L-2}{l+i}\alpha_i+$ $\sum_{j=1}^{l}\binom{L-2}{j-1}\beta_j$
0	$p_{01}=\sum_{i=0}^{l-1}\binom{L-2}{l+i}\alpha_i+$ $\sum_{j=1}^{l}\binom{L-2}{j-1}\beta_j$	$p_{00}=\sum_{i=0}^{l-2}\binom{L-2}{l+1+i}\alpha_i+$ $\sum_{j=0}^{l}\binom{L-2}{j}\beta_j$

The relationship between any two classifiers f_i and f_k can be visualized using a pairwise table. In Table 3.1, the probability of every correct/incorrect combination is presented, where "1" stands for the correct vote and "0" stands for the incorrect vote. The entries in the table are obtained by following combinatorial reasoning. For example, when both f_i and f_k are correct, the remaining $(L-2)$ classifiers either give $(l-1+i)$ correct votes with $\binom{L-2}{l-1+i}$ ways if the majority vote is correct, or give $(j-2)$ correct votes with $\binom{L-2}{j-2}$ ways if the majority vote is wrong. Thus, the probability of having f_i and f_k both correct is $\sum_{i=0}^{l}\binom{L-2}{l-1+i}\alpha_i+\sum_{j=2}^{l}\binom{L-2}{j-2}\beta_j$. Based on the assumption of all individuals holding the same accuracy p, the pattern is symmetrical with respect to all classifiers, so that all pairs of individual classifiers have the same pairwise tables, and therefore the same Q. According to the definition of Q-statistic, it can be computed by substituting the four probabilities in Eq. 3.8 with the expressions in Table 3.1,

$$Q=\frac{p_{11}p_{00}-p_{01}p_{10}}{p_{11}p_{00}+p_{01}p_{10}}. \tag{3.8}$$

Due to too many α and β terms involved in Q, we derive an upper bound on Q-statistic to simplify our discussions here.

Theorem 3.3. *Q-statistic is upper bounded by the monotone decreasing functions of* p_{01} *(*p_{10}*) under the assumption of individual classifiers having the same accuracy* p.

Proof. Due to the same individual accuracy p, $p_{01} = p_{10}$ holds. Q-statistic can be expressed as

$$Q = \frac{p_{00}p_{11} - p_{01}^2}{p_{00}p_{11} + p_{01}^2}.$$

Because the inequality $p_{00}p_{11} \leq \left(\frac{p_{00}+p_{11}}{2}\right)^2$ holds and the four probabilities satisfy $p_{00} + p_{11} + p_{01} + p_{10} = 1$ ($p_{01} < 1/2$), we obtain

$$Q \leq \begin{cases} \frac{1-4p_{01}}{4p_{01}^2} & \text{if } Q \geq 0 \\ \frac{1-4p_{01}}{4p_{01}^2+(1-2p_{01})^2} & \text{if } Q < 0 \end{cases}$$

by eliminating p_{00} and p_{11}. The right-hand side functions of the inequality are monotone decreasing with respect to p_{01} in $[0, 1/2]$, which upper-bound Q. □

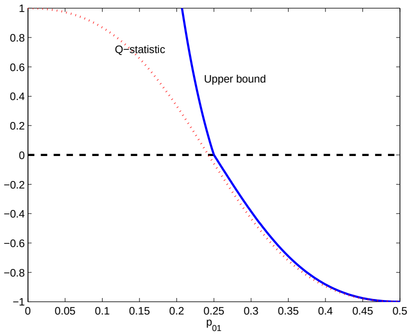


Figure 3.1: Q-statistic and its upper bound with respect to p_{01}.

It is easy to understand Theorem 3.3 intuitively. It tells us that increasing the probability of making different decisions of pairs of classifiers can make the ensemble more diverse. Fig. 3.1 plots the curves of Q-statistic and its upper bound with respect to p_{01}. When p_{01} is very small, the upper bound is much larger than the Q-statistic value and less useful. As p_{01} gets larger, the upper bound becomes tighter. $p_{01} > \frac{1}{4}$ guarantees a negative Q. Therefore, by increasing p_{01}, Q-statistic can be upper-bounded better and a more diverse ensemble will be achieved. Now, we just concentrate on the expression of p_{01} instead of Q, and see how it relates to P_{ovr} through the voting distribution of the ensemble. It is reformulated in Eq. 3.9.

$$
\begin{aligned}
p_{01} = & \underbrace{(\binom{L-2}{l-1}\alpha_0}_{\text{best}} + \binom{L-2}{l-2}\alpha_1 \ldots + \binom{L-2}{0}\alpha_{l-1}) + \\
& \underbrace{(\binom{L-2}{l-1}\beta_l}_{\text{worst}} + \binom{L-2}{l-2}\beta_{l-1} \ldots + \binom{L-2}{0}\beta_1)
\end{aligned}
\tag{3.9}
$$

Now we consider the best and worst patterns from the view of the general pattern. According to the definition of the best pattern, p_{01} is reduced to $\binom{L-2}{l-1}\alpha_0$ as denoted in Eq. 3.9. Increasing p_{01} means raising the probability of making the correct prediction of the ensemble, in that only the α term exists. Similarly, $p_{01} = \binom{L-2}{l-1}\beta_l$ in the worst pattern. Increasing p_{01} leads to higher probability of making the wrong prediction, and thus worse overall accuracy. *What does it suggest?* If α terms dominate p_{01}, then P_{ovr} is positively related to p_{01}, and encouraging diversity causes a performance improvement. Otherwise, P_{ovr} is negatively related to p_{01} by β terms, and higher diversity reduces the accuracy. Accordingly, we infer that the number of wasted correct votes of the ensemble determines which types of voting combinations would be positively affected more by diversity. Increasing diversity tends to make α-type voting combinations more likely to happen when few correct votes are wasted. The analysis here helps us to understand the impact of diversity on single-class performance for class imbalance problems next.

Before diverting our attention, we clearly define good and bad patterns for the following discussions. In general, if increasing diversity causes higher α's and lower β's (i.e. better accuracy), we say that the ensemble has a *good voting pattern*; if the case of lower α's and higher β's happens (i.e. worse accuracy), it has a *bad voting pattern.*

3.2.3 Patterns and Class Imbalance Learning

In this section, we take the classification characteristics of class imbalance learning into consideration. We first give some insight into the class imbalance problem from the view of base learning algorithms, such as decision trees and neural networks that are commonly used to form an ensemble. For these kinds of learners, the classification difficulty caused by skewed class distributions and different misclassification costs is mainly reflect in the overfitting to the minority class and the over-generalization to the majority class, because the small class has less contribution to the classifier. Both decision trees and neural networks have been reported to be biased toward the majority concept inherently (He and Garcia, 2009; Weiss, 2004; Japkowicz and Stephen, 2002). A tree learner can result in very specific branches for the minority class that cover very few training examples. A neural network cannot learn the minority class sufficiently, because the majority class examples overwhelm the minimization procedure of the squared error using gradient descent.

Considering an ensemble composed of many of such classifiers, each classifier tends to label most of the data as the majority class. We can imagine that the ensemble has a very low diversity. As an extreme situation, all individuals misclassify any minority class example and assign the majority class label to all examples. Based on this understanding, we propose two arguments for each class from the view of patterns.

(1) Minority class and good pattern: Recall the general pattern in the previous section, where Q-statistic and the overall accuracy are linked through p_{01} expressed by α and β terms. For the minority class, each individual classifier has a low recognition rate. It corresponds to large β and small α values in the general pattern. Moreover, very little disagreement degree among the individuals suggests that the number of wasted correct votes is small. Increasing p_{01} in this situation is prone to cause larger α's and smaller β's. We hence argue that the ensemble tends to have a good pattern over the minority class, where encouraging diversity can improve its classification accuracy.

(2) Majority class and bad pattern: The majority class contains sufficient data information for learners. Consequently, every individual tends to make the same correct decision. Referring to the general pattern, it means large α and small β values, plus a lot of wasted correct votes from the ensemble. Increasing p_{01} in this situation is very likely to reduce α's and increase β's. In this regard, the ensemble tends to behave in a bad pattern over

the majority class, where diversity deteriorates the accuracy.

It is worth noting that the accuracy here is in the context of a single class. The two arguments reflect the fact of classifying an imbalanced data based on the understanding of base classifiers. Different effects of diversity between classes are expected in imbalanced scenarios. The empirical evidence of how diversity affects the performance in each type of classes will be given in section 3.4.

3.3 Relationship Between Diversity and Single-Class Measures

The relationship discussed earlier is concerned with overall accuracy of majority vote. We extend it to single-class performance in this section. Three frequently used single-class measures are included here as we have reviewed in section 2.1.3: recall, precision and F-measure. We will show how they behave in two extreme patterns as Q-statistic varies. Six possible situations are obtained through mathematical analysis. It should be noted that subscripts "min" and "maj" appearing in this section just stand for two classes of a data set in general without discriminating the class size, so the obtained results of the impact of diversity on single-class performance are applicable to both classes here.

For any input x_j, two probabilities can be approximated: $p\left\{(x_j, y_j) \in Z_{min}\right\} = \theta$ and $p\left\{(x_j, y_j) \in Z_{maj}\right\} = 1 - \theta$, where again θ is the proportion of Z_{min} in Z. They are constant values for a given problem. We denote two additional probabilities as follows,

$$p_{tp} = p\left\{O_{j,ens} = 1 \cap (x_j, y_j) \in Z_{min}\right\}, \tag{3.10}$$

$$p_{fp} = p\left\{O_{j,ens} = 0 \cap (x_j, y_j) \in Z_{maj}\right\}. \tag{3.11}$$

$O_{j,ens}$ is the oracle output of the ensemble following the definition in section 2.3.1. p_{tp} indicates the probability of any minority class example predicted correctly by the ensemble. p_{fp} is the probability of any majority class example misclassified by the ensemble. According to the definitions of recall and precision, they can be re-expressed by

$$R_{min} = \frac{p_{tp}}{\theta}, \tag{3.12}$$

$$P_{min} = \frac{p_{tp}}{p_{tp} + p_{fp}}. \tag{3.13}$$

3.3.1 Recall, Precision and F-measure with Independence Assumption

We consider a simple case first. We assume whether an input z_j is classified correctly by the ensemble is independent of its real class label. Under the assumption, Eq. 3.10 and Eq. 3.11 can be simplified into

$$p_{tp} = \theta P_{ovr}, \tag{3.14}$$

$$p_{fp} = (1 - \theta)(1 - P_{ovr}). \tag{3.15}$$

Substituting them into Eq. 3.12 and Eq. 3.13, we obtain

$$R_{min} = P_{ovr}, \tag{3.16}$$

$$P_{min} = \frac{\theta P_{ovr}}{\theta P_{ovr} + (1 - \theta)(1 - P_{ovr})}. \tag{3.17}$$

Eq. 3.16 and Eq. 3.17 only contain P_{ovr} and the constant θ. By substituting P_{ovr} by the monotonic functions in the best and worst patterns, we obtain clear links of recall and precision to Q-statistic. The recall measure has the same functional relation as P_{ovr}, which is monotone decreasing with respect to Q in the best pattern and monotone increasing in the worst pattern. As to precision in the best pattern,

$$P_{min} = \frac{-Q(L\theta) + L\theta}{-Q(3l\theta + \theta - l) + (2l\theta + 1)}. \tag{3.18}$$

It is a monotone decreasing function with respect to Q, because its derivative is always negative. In other words, increasing diversity can improve both recall and precision in the best pattern, and thus a better F-measure is obtained.

In the worst pattern,

$$P_{min} = \frac{Q\,(l\theta)\,(l-1) + \theta\,(5l+1)}{Q\,(3l\theta + \theta - L)\,(l-1) + (2l^2 - 2l^2\theta + 10l\theta - 5l + 4\theta - 3)}. \tag{3.19}$$

It is a monotone increasing function with respect to Q. R_{min} and P_{min} will increase as Q increases in the worst pattern. Therefore, enforcing diversity leads to the reduction of recall, precision and F-measure in this case.

In summary, Q-statistic has the same impact on the single-class measures in each pattern under the independence assumption. Recall, precision and F-measure get improved or reduced simultaneously depending on "good" or "bad" Q. For class imbalance problems, however, this assumption is hard to hold, since the minority class data are more likely to be misclassified in general than the data belonging to the majority class. As the data set gets less imbalanced, the dependence between the class label and misclassification should get smaller. To certain extent, we can regard this part of discussions to be suitable for balanced data sets.

3.3.2 Recall, Precision and F-measure without Independence Assumption

Without the above assumption, it is not easy to get such neat and separate expressions for those single-class measures. They are related to each other and can behave in different ways. Multiple situations must be considered. The overall accuracy (P_{ovr}) is utilized to associate Q-statistic with single-class measures here. P_{ovr} can be re-expressed by every two single-class measures according to their definitions through some mathematical transformations:

$$P_{ovr} = \frac{TP + TN}{|Z|} = \frac{|Z_{min}| \cdot R_{min} + |Z_{maj}| \cdot R_{maj}}{|Z|}. \tag{3.20}$$

The single-class measures in Eq. 3.20 are from different classes. To express P_{ovr} with measures from the same class, we use Eq. 3.12 divided by Eq. 3.13, and get

$$\frac{R_{min}}{P_{min}} = R_{min} + \frac{1-\theta}{\theta}\left(1 - R_{maj}\right). \tag{3.21}$$

Eq. 3.21 presents the relation among R_{min}, P_{min} and R_{maj}. Eliminating R_{maj} from Eq. 3.20 and Eq. 3.21, we obtain

$$P_{ovr} = (1-\theta) + \theta\left(2 - \frac{1}{P_{min}}\right) R_{min}. \tag{3.22}$$

Similarly,

$$P_{ovr} = 1 + 2\theta R_{min}\left(1 - \frac{1}{F_{min}}\right). \tag{3.23}$$

$$P_{ovr} = (1-\theta) + \theta\left(\frac{2P_{min} - 1}{\frac{2P_{min}}{F_{min}} - 1}\right). \tag{3.24}$$

P_{ovr} is expressed by every two single-class measures from one class. It is monotone increasing with respect to $R_{min}/P_{min}/F_{min}$. The changing behaviours of the measures are concluded in Table 3.2 based on the above functional relations, including six possible situations in good and bad patterns in terms of overall accuracy. In situation (3), for example, if the ensemble performs in a good pattern (i.e. decreasing Q leads to better P_{ovr}) and R_{min} decreases, then P_{min} will increase due to Eq. 3.22 and F_{min} will increase due to Eq. 3.23. Any two single-class measures from the same class will not get worse simultaneously in the good pattern according to the links. Analogously, any two single-class measures from the same class will not increase simultaneously in the bad pattern. Table 3.2 is also applicable to $R_{maj}/P_{maj}/F_{maj}$. The mathematical functions of the three single-class measures derived in section 3.3.1 fall into situations (1) and (6) as special cases.

The table answers the question of the relationship between ensemble diversity and

Table 3.2: The possible changing direction of R_{min}, P_{min} and F_{min} as Q-statistic decreases (i.e. increasing diversity) in "good" and "bad" patterns in terms of P_{ovr}.

$Q \downarrow$	P_{ovr}	R_{min}	P_{min}	F_{min}	Situation
Overall good pattern	↑	↑	↑	↑	(1)
			↓	↑	(2)
		↓	↑	↑	(3)
Overall bad pattern	↓	↑	↓	↓	(4)
		↓	↑	↓	(5)
			↓	↓	(6)

single-class measures through theoretical links, but it is still unclear about which situation would happen in class imbalance scenarios. The next questions are: *how does diversity affect the classification performance of the minority and majority classes in real imbalanced scenarios? Which situation does the ensemble have over each class?* According to our arguments in section 3.2.3, the impact of diversity tends to be different between classes. It is expected to be beneficial to the recognition of minority class examples, but may cause a negative effect to the majority class. Hence, different situations are anticipated.

3.4 Diversity Analysis for Class Imbalance Learning

To verify the obtained results so far, we examine the relationship between diversity and classification performance empirically in class imbalance scenarios in this section. Artificial and highly imbalanced real-world benchmark data sets are included in the experiments.

3.4.1 Impact of Diversity on Classification Performance on Artificial Imbalanced Data

To clearly observe the impact of diversity on balanced and imbalanced data sets and investigate how the performance measures behave as diversity varies, we build ensembles on three two-dimensional artificial data sets. We proceed with correlation analysis to show the relationship between Q-statistic and other performance measures, and present corresponding decision boundary plots to explain the obtained results. The correlations are further confirmed by discussing another non-pairwise diversity measure – generalized diversity (GD). Then, we examine if the findings are also applicable to imbalanced data with a larger training size and a larger feature space.

Experimental Setup

Artificial data is generated from two Gaussian distributions with equal covariance and a small overlapping area close to the separating line. Three different data sizes are considered: one class always contains 200 training points, while the size of the other class is set to 10 (very imbalanced), 50 (imbalanced) and 200 (balanced) respectively. They are denoted by "200-10", "200-50" and "200-200". By applying the same method, a corresponding testing file is created with 50 points in each class.

As we know, Bagging (Breiman, 1996) achieves diversity through resampling, where each training subset is kept different from each other. In our experiments, we apply the Bagging training strategy and manipulate diversity by tuning the sampling rate $r\%$. A smaller r means that fewer points join the training. Each training subset is less likely to be similar to each other. Thus, the prediction becomes less stable and a larger diversity degree is expected (Skurichina et al., 2002). Different from the conventional Bagging, a different sampling rate is applied to each class of the imbalanced data set in our experiments. Concretely, the majority class is randomly sampled with replacement at rate $r\%$; the minority class is randomly sampled with replacement at rate $r\left(1-\theta\right)/\theta\%$, where θ is the imbalance rate as defined before. By doing so, every training subset has a balanced class distribution. This is to avoid the situation that the minority class is ignored by the classifier, and thus affecting our experimental analysis with misleading results. Some preliminary experiments showed that the recall of the minority class always remains 0 without rebalancing applied when the training data is highly skewed, which means that no minority example is recognized. The training strategy is described in Table 3.3. A simple majority vote gives the final decision.

Table 3.3: Bagging-based training strategy (Wang and Yao, 2009b).

```
Training:
1. Given the training set $Z$ with imbalance rate $\theta$;
   resampling rate at $r\%$; number of classifiers $L$.
   $Z_{min}$ is the minority class subset of $Z$.
   $Z_{maj}$ is the majority class subset of $Z$.
2. Construct subset $Z_k$ by executing the following:
   2a. Bootstrap $Z_{maj}$ at rate $r\%$
       and add chosen examples into $Z_k$;
   2b. Bootstrap $Z_{min}$ at rate $r\left(1-\theta\right)/\theta\%$
       and add chosen examples into $Z_k$.
3. Train a classifier using $Z_k$.
4. Repeat steps 2 and 3 until $k$ equals $L$.

Testing on a new example: (majority voting)
1. Collect decisions from each classifier.
2. Return the class label that receives the most votes.
```

The training method is run 50 times for each setting of sampling rate, and the averages are computed. The sampling rate $r\%$ is varied in the range of $[3\%, 1000\%]$. Every ensemble consists of 15 classifiers. C4.5 decision trees are used as the base learner. For the correlation analysis, we output 10 measures, including overall Q-statistic (Q), minority-class Q-statistic (Q_{min}), majority-class Q-statistic (Q_{maj}), minority-class recall (R_{min}), minority-class precision (P_{min}), minority-class F-measure (F_{min}), majority-class recall (R_{maj}), majority-class precision (P_{maj}), majority-class F-measure (F_{maj}) and overall accuracy (P_{ovr}). Q_{min} and Q_{maj} assess the diversity degree of an ensemble only within the minority and majority data subsets respectively.

Correlation Analysis of Q-statistic

We conduct correlation analysis to study the relationship between diversity and single-class performance. We compute Spearman's rank correlation coefficient between overall diversity Q and the other nine measures. It is a non-parametric measure of statistical dependence between two variables, and insensitive to how the measures are scaled. It ranges in $[-1, 1]$, where 1 (or -1) indicates a perfect monotone increasing (or decreasing) relationship. Table 3.4 summarizes the correlation coefficients in two sampling ranges. A significance correlation is indicated in boldface at confidence level of 95%.

Table 3.4: Rank correlation coefficient (in %) between overall diversity Q and 9 measures on 3 artificial data sets. Numbers in boldface indicate significant correlations.

$r \in [3, 100]$	Q_{min}	Q_{maj}	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
200-200	**93**	**52**	27	**-54**	-9	**-50**	26	-22	-16
200-50	**97**	**98**	**-98**	**54**	**-98**	**58**	**-98**	**-98**	**-98**
200-10	6	**98**	**-38**	**33**	**-36**	**33**	**-36**	**-36**	**-36**
$r \in [100, 1000]$	Q_{min}	Q_{maj}	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
200-200	**98**	**89**	**-79**	**-89**	**-98**	**-81**	**-88**	**-98**	**-96**
200-50	**100**	**93**	**-81**	**58**	**-81**	**58**	**-81**	**-81**	**-81**
200-10	**99**	27	**-81**	**77**	**-61**	**77**	**-55**	**-55**	**-61**

First, between Q and Q_{min}/Q_{maj}, we observe that all coefficients from the three data sets are positive, which shows that ensemble diversity for each class has the same changing tendency as the overall diversity, regardless of whether the data set is balanced. On one hand, it guarantees that increasing the classification diversity over the whole data set can increase diversity over each class. On the other hand, it confirms that the diversity measure Q-statistic seems not to be sensitive to imbalanced distributions.

For the balanced data set "200-200", the overall accuracy and most single-class measures do not present clear correlations with Q when $r \in [3, 100]$. In this range, increasing diversity does not necessarily lead to better performance, due to the mutual effect of individual accuracy and diversity. When r varies in $[100, 1000]$ with a low diversity range, all 7 performance measures have strong negative correlations with Q. It suggests that diversity is beneficial to both classes and the overall accuracy. This observation agrees with the mathematical relations under the independence assumption in section 3.3.1. It corresponds to the situation 1 in Table 3.2. Generally speaking, when ensemble diversity is small, increasing diversity improves the classification performance of both classes in the balanced case.

Imbalanced data sets "200-50" and "200-10" present similar impacts of diversity on each performance measure and more significant correlations than the balanced data set. Overall accuracy P_{ovr} gets higher in both ranges of r as the ensemble becomes more diverse. Single-class measures behave differently between classes. With respect to the minority class, R_{min} and F_{min} have significant negative correlations with Q; P_{min} has a significant positive correlation with Q. It implies that increasing diversity can find more minority class examples (i.e. better recall) but lose some classification precision, due to the trade-off between recall and precision. Better F-measure indicates that the improvement of recall is greater than the reduction of precision. The observation corresponds to the situation 2 in Table 3.2. As to the majority class, R_{maj} has a significant positive correlation with Q; P_{maj} and F_{maj} have significant negative correlations with Q. It means that the majority-class recall gets smaller along with the increase of diversity, but precision and F-measure are improved. The measure behaviours of the majority class correspond to the situation 3 in Table 3.2. It implies the performance trade-off between minority and majority classes.

Table 3.5 summarizes the measure tendencies for both balanced and imbalanced cases along with the decrease of Q (i.e. the increase of ensemble diversity) and their corresponding situations in Table 3.2.

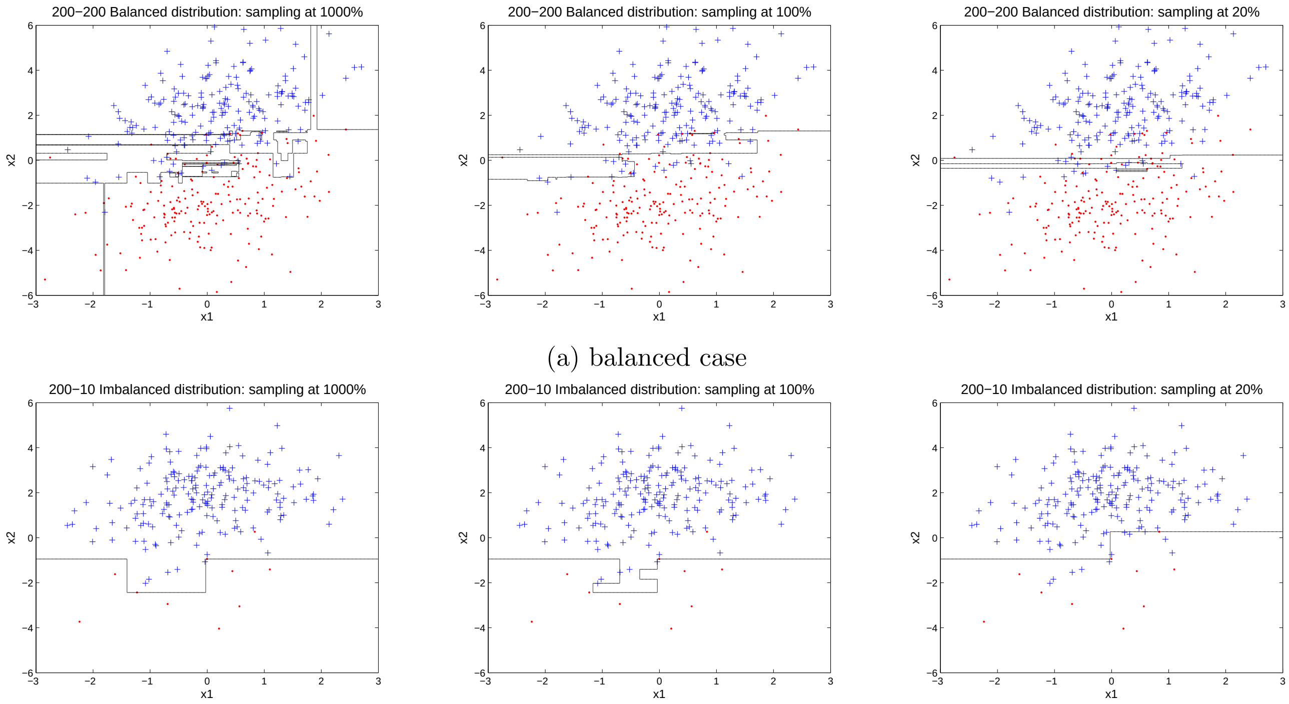
Table 3.5: Changing directions of single-class measures and overall accuracy by increasing diversity on artificial data sets and their corresponding situations in Table 3.2.

Class	P_{ovr}	R	P	F	Situation
Single-class (200-200)	↑ (good pattern)	↑	↑	↑	(1)
Minority (200-50,200-10)	↑ (good pattern)	↑	↓	↑	(2)
Majority (200-50,200-10)		↓	↑	↑	(3)

Considering the learning objective in class imbalance learning and the importance of the minority class, we conclude that diversity has a positive effect in classifying these imbalanced data sets. The performance of both classes is better balanced between recall and precision with more minority class examples identified. Moreover, the different behaviours of recall between the minority and majority classes agree with the two arguments in section 3.2.3.

Decision Boundary Analysis

To show the radical effects of diversity visually, we produce classification boundary plots for data sets “200-200” and “200-10” at three specific sampling rates in Fig. 3.2: $r = 1000$ with a low diversity degree, $r = 100$ with a moderate diversity degree, and $r = 20$ with a high diversity degree.



(a) balanced case

(b) imbalanced case

Figure 3.2: Classification boundary plots produced by Bagging-based tree ensembles at sampling rates 1000%, 100%, and 20% on data sets “200-200” (balanced) and “200-10” (very imbalanced).

For the balanced case in Fig. 3.2(a), we can see that the main effect of diversity is to make the ensemble less overfit the training data close to boundaries. Hence, diversity improves the performance of both classes. For the imbalanced case in Fig. 3.2(b), in addition to a less overfitting boundary, diversity expands it towards the majority class side. A broader boundary is obtained for the minority class. It explains why more minority class examples are identified with majority-class recall sacrificed to some extent. Table 3.6 presents the raw outputs of Q-statistic and the other performance measures for the above cases, to give a clear idea of how they change at different sampling rates.

Table 3.6: Means of Q-statistic and the other performance measures on data sets "200-200" and "200-10" at sampling rates 1000%, 100% and 20%.

200-200	Q	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
1000%	0.982	0.877	0.958	0.916	0.961	0.887	0.922	0.919
100%	0.922	0.908	0.975	0.941	0.977	0.914	0.945	0.943
20%	0.819	0.903	0.985	0.943	0.985	0.907	0.944	0.943
200-10	Q	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
1000%	0.977	0.571	1.000	0.725	1.000	0.701	0.824	0.785
100%	0.974	0.659	1.000	0.794	1.000	0.746	0.855	0.830
20%	0.960	0.836	1.000	0.911	1.000	0.860	0.924	0.918

Correlation Analysis of Non-Pairwise Diversity Measure

To confirm our results of the impact of diversity measured by Q-statistic, we carry out the same correlation analysis based on a popular non-pairwise diversity measure – generalized diversity (GD), which has been defined in chapter 2. It assumes that the maximum diversity occurs when misclassification of one classifier is accompanied by correct prediction from the other classifier, and minimum diversity occurs when one's misclassification is always accompanied by failure of the other. Unlike Q-statistic where a small value implies a more diverse ensemble, higher GD indicates greater diversity. Spearman's rank correlation coefficients between GD and the other performance measures are computed. The results are shown in Table 3.7.

Similar observations are obtained compared to Table 3.4: 1) for the balanced data set, not all correlations with GD are significant when r ranges in $[3, 100]$. In the higher sampling range of $[100, 1000]$, all 7 performance measures are positively correlated with GD, which shows the positive role of diversity in both classes. 2) for the two imbalanced

Table 3.7: Rank correlation coefficient (in %) between GD and the other performance measures on artificial data sets “200-200”, “200-50” and “200-10”. Numbers in boldface indicate significant correlations.

$r \in [3, 100]$	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
200-200	-17	**51**	27	**47**	-9	**35**	**31**
200-50	**99**	**-40**	**99**	**-48**	**99**	**99**	**99**
200-10	**72**	6	**72**	6	**72**	**72**	**72**
$r \in [100, 1000]$	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
200-200	**67**	**94**	**98**	**93**	**73**	**98**	**98**
200-50	**96**	**-68**	**99**	**-70**	**96**	**96**	**96**
200-10	**40**	**-50**	**40**	**-50**	**39**	**39**	**39**

data sets, increasing diversity improves R_{min}, F_{min}, P_{maj}, F_{maj} and P_{ovr} accompanied by some reduction of P_{min} and R_{maj}. It shows that diversity is beneficial to the performance of both classes in terms of F-measure and more minority class examples are identified.

Larger Training Data and Larger Feature Space

We have investigated the impact of ensemble diversity on single-class performance measures in depth on three small artificial data sets. To obtain a complete understanding, we consider imbalanced training data with a larger size and a larger feature space here. Two more training data sets are generated and discussed: “2000-100” data set with 2 dimensions, where the majority class contains 2000 examples and the minority class contains 100; “200-10” data set with 20 dimensions, where the majority class contains 200 examples and the minority class contains 10. Each dimension follows a Gaussian distribution. The correlation coefficients of the single-class measures and overall accuracy with both Q-statistic and GD are calculated and shown in Tables 3.8 - 3.9.

For data set “2000-100”, the results in Table 3.8 show that increasing diversity reduces R_{min}, F_{min}, P_{maj}, F_{maj} and P_{ovr} when $r \in [3, 100]$, which is opposite to the observation in the earlier case of the small training set “200-10” with 2 dimensions. When $r \in [100, 1000]$, diversity presents a positive impact on them, which also happens to the small training set. It suggests that diversity may not always benefit the minority class if the training data

Table 3.8: Rank correlation coefficient (in %) between Q/GD and the other performance measures on data set "2000-100" with 2 dimensions. Numbers in boldface indicate significant correlations.

Q	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
$r \in [3, 100]$	**74**	**-48**	**74**	**-48**	**74**	**75**	**75**
$r \in [100, 1000]$	-23	**42**	-22	**42**	-22	-22	-22
GD	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
$r \in [3, 100]$	**-89**	**68**	**-89**	**68**	**-89**	**-88**	**-88**
$r \in [100, 1000]$	**87**	**67**	**87**	**67**	**87**	**87**	**87**

is large enough. Making full use of available training data seems to be more important when the ensemble keeps certain degree of diversity. Diversity plays a better role when the ensemble has a relatively low diversity degree.

Table 3.9: Rank correlation coefficient (in %) between Q/GD and the other performance measures on data set "200-10" with 20 dimensions. Numbers in boldface indicate significant correlations.

Q	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
$r \in [3, 100]$	**-82**	**79**	-2	**79**	**-68**	16	8
$r \in [100, 1000]$	**-78**	**-77**	**-78**	**-77**	**-77**	**-77**	**-78**
GD	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
$r \in [3, 100]$	**83**	**-76**	4	**-77**	**70**	-13	-5
$r \in [100, 1000]$	**85**	**84**	**84**	**84**	**84**	**84**	**85**

For data set "200-10" with 20 dimensions, the imbalanced data are more sparsely distributed in the larger feature space and overfitting is more likely to happen than the data set "200-10" with only 2 dimensions. As shown in Table 3.9, diversity appears to be useful for the high-dimensional imbalanced case. Increasing diversity improves R_{min} in both ranges of r, so more minority class examples are identified. Diversity is beneficial to precision and F-measure of both minority and majority classes in the higher range of r. When r varies in the smaller range, P_{min} is decreased by increasing diversity, and F-measure and overall accuracy do not present significant correlations.

3.4.2 Impact of Diversity on Classification Performance on Real-World Imbalanced Data

So far, we have shown the positive impact of diversity on single-class performance in depth through artificial data sets. Now we ask whether the results are applicable to real-world domains. In this section, we report the results for the same research question on fifteen highly imbalanced real-world benchmarks. The data information is summarized in Table 3.10. Particularly, the first ten data sets come from the PROMISE data repository (Boetticher et al., 2007), which are collected from real software engineering projects. The task of this group of data is to predict defects existing in programming codes. There are two classes in each data set, defect and non-defect. They are highly imbalanced in nature, because defect examples are much less likely to occur than non-defect ones. The "insurance" data set comes from CoIL data mining competition (Putten and Someren, 2004). The goal is to predict who would be interested in buying a specific insurance product. The last four data sets are chosen from the UCI repository (Frank and Asuncion, 2010) that are frequently discussed in the class imbalance learning literature. It is worth mentioning that the UCI data sets originally have more than two classes. We adapt them into two-class data problems by selecting a small class as the minority and merging the others as the majority.

The same Bagging-based training strategy is applied as in the previous subsection (Table 3.3). The relationship is studied through correlation analysis using Spearman's rank correlation coefficient. The sampling rate $r\%$ is varied in $[3\%, 100\%]$ with interval 2. At each sampling rate, 15 decision trees are built to form an ensemble. "Insurance" has a separate testing file, and each setting is repeated 30 times. A 5-fold cross validation is applied to the other data sets with 10 runs.

Single-Class Performance

Table 3.11 shows the correlations between diversity measured by Q and GD and the other performance measures for each data set, including three single-class measures $R/P/F$ of

Table 3.10: Summary of real-world data sets.

Data	Size	Attributes	Imbalance Rate
mc2	161	39	32.29%
mw1	403	37	7.69%
kc3	458	39	9.38%
cm1	498	21	9.83%
kc2	522	21	20.49%
pc1	1109	21	6.94%
pc4	1458	37	12.20%
pc3	1563	37	10.23%
kc1	2109	21	15.45%
pc2	5589	36	0.41%
insurance	5822	85	5.98%
glass	214	9	7.94%
ecoli	336	7	10.42%
balance	625	4	7.84%
car	1728	6	3.99%

both classes and overall accuracy P_{ovr}. Strong correlations are found.

Different from the artificial cases, diversity harms the overall accuracy according to its significant positive correlation with Q in 14 data sets and negative correlation with GD in 13 data sets. Larger diversity causes worse P_{ovr}. Referring it to the pattern analysis in section 3.2, an ensemble tends to perform in a good pattern in artificial scenarios, whereas it's more likely to behave in a bad pattern in real-world imbalanced domains. The reason could be the more complex data distributions and noisy examples in practical tasks.

More importantly, single-class performance should be the focus. For the minority class, recall has a very strong negative correlation with Q in all cases and a strong positive correlation with GD in 14 cases; precision has a very strong positive correlation with Q in 12 cases and a strong negative correlation with GD in 11 cases; the coefficients of F-measure do not show a consistent relationship. The observation suggests that more minority-class examples are identified with some loss of precision by increasing diversity. When the improvement of recall is larger than the drop of precision, F-measure gets better. Otherwise, it gets worse. Both directions could happen to F_{min}. Their behaviours

Table 3.11: Rank correlation coefficient (in %) between diversity measured by Q/GD and the other performance measures on real-world imbalanced data sets. Numbers in boldface indicate significant correlations.

Q	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
mc2	**-86**	**70**	-11	**89**	**-40**	**84**	**76**
mw1	**-96**	**86**	**31**	**98**	**-94**	**97**	**96**
kc3	**-98**	**96**	**31**	**99**	**-96**	**99**	**99**
cm1	**-98**	**59**	**-89**	**99**	**-97**	**98**	**98**
kc2	**-98**	-4	**-92**	**94**	**-97**	**59**	10
pc1	**-99**	**96**	**47**	**99**	**-98**	**98**	**98**
pc4	**-97**	**94**	8	**96**	**-97**	**93**	**93**
pc3	**-99**	**97**	-22	**99**	**-98**	**99**	**99**
kc1	**-99**	**97**	-7	**99**	**-99**	**99**	**98**
pc2	**-62**	-12	**-34**	**83**	**-62**	**82**	**82**
insurance	**-100**	**97**	**-72**	**100**	**-99**	**100**	**100**
glass	**-88**	**83**	**43**	**95**	**-84**	**93**	**93**
ecoli	**-77**	**77**	**48**	**80**	**-77**	**80**	**79**
balance	**-77**	**-77**	**-77**	**90**	**31**	**90**	**90**
car	**-84**	**81**	**66**	**83**	**-84**	**75**	**74**

GD	R_{min}	P_{min}	F_{min}	R_{maj}	P_{maj}	F_{maj}	P_{ovr}
mc2	**78**	**-56**	4	**-81**	**38**	**-68**	**-53**
mw1	**95**	**-83**	23	**-92**	**95**	**-92**	**-91**
kc3	**80**	**-73**	**46**	**-82**	**79**	**-81**	**-80**
cm1	15	1	**31**	-15	16	-15	-15
kc2	**90**	**51**	**92**	**-83**	**90**	-19	**45**
pc1	**96**	**-80**	24	**-92**	**95**	**-88**	**-87**
pc4	**78**	**-57**	**80**	**-72**	**78**	**-56**	**-51**
pc3	**83**	**-82**	**74**	**-89**	**83**	**-88**	**-87**
kc1	**96**	**-89**	**81**	**-95**	**96**	**-93**	**-92**
pc2	**33**	12	19	**-86**	**29**	**-86**	**-86**
insurance	**97**	**-95**	**96**	**-97**	**97**	**-97**	**-97**
glass	**91**	**-88**	**-53**	**-92**	**86**	**-92**	**-91**
ecoli	**84**	**-66**	6	**-74**	**83**	**-68**	**-65**
balance	**75**	**75**	**75**	**-88**	**-67**	**-88**	**-88**
car	**86**	**-81**	**-74**	**-83**	**86**	**-81**	**-81**

correspond to the situation 4 in Table 3.2, except that F_{min} does not consistently follow the mathematical link in practice. For the majority class, recall and F-measure have very strong positive correlations with Q and negative correlations with GD; precision has a very strong negative correlation with Q and a positive correlation with GD. It indicates that although diversity helps to predict majority class examples more accurately, recall is sacrificed more than the improvement of precision. The result corresponds to the situation 5 in Table 3.2.

Generally speaking, diversity is helpful for recognizing minority class examples in real-world scenarios. A better balance between recall and precision of the minority class is achieved in partial cases. Diversity degrades the classification performance of the majority class in terms of recall and F-measure. The behaviours of recall of two classes tally with the arguments in section 3.2.3. Table 3.12 summarizes the measure behaviours along with the increase of ensemble diversity and the corresponding situations of each class in Table 3.2.

Table 3.12: Changing directions of single-class measures and overall accuracy by increasing diversity on 15 real-world data sets and their corresponding situations in Table 3.2. '-' indicates an unclear changing behaviour obtained.

Class	P_{ovr}	R	P	F	Situation
Minority	↓	↑	↓	-	(4)
Majority	(bad pattern)	↓	↑	↓	(5)

Overall Performance

We have analyzed the correlations of overall accuracy and single-class performance measures with respect to diversity. *Why do we need to discuss overall performance here?* As we have explained in previous chapters, accuracy is not a good overall performance measure for class imbalance problems, which is strongly biased to favor the majority class. Although the single-class measures reflect better the performance information for one class, it is still necessary to evaluate how well a classifier can balance the performance between classes.

Class imbalance learning uses better indicators to show the performance trade-off between classes. As overall performance measures, G-mean (Kubat et al., 1997; Kubat and Matwin, 1997) and AUC (Swets, 1988; Bradley, 1997) are most widely used and insensitive to class distributions (He and Garcia, 2009). G-mean is the geometric mean of recalls of the minority and majority classes, which is defined as $\sqrt{R_{min} \cdot R_{maj}}$. It shows that to what extent accuracy on the majority class drops with the error reduction on the minority class. AUC is the area under the ROC curve, a two-dimensional graph between TP rate and FP rate. The best algorithm should produce the dominant curve, which also has the largest AUC. It has been proved that AUC is statistically consistent and more discriminating than accuracy (Ling et al., 2003).

Table 3.13: Rank correlation coefficient (in %) between diversity measured by Q/GD and overall performance measures including G-mean and AUC on real imbalanced data sets. Numbers in boldface indicate significant correlations.

data	Q-statistic		GD	
	G-mean	AUC	G-mean	AUC
mc2	5	**60**	-3	**-36**
mw1	**-94**	**-61**	**91**	**90**
kc3	**-97**	**-76**	**80**	**91**
cm1	**-96**	**-68**	16	**44**
kc2	**-95**	**-96**	**90**	**91**
pc1	**-97**	**-70**	**95**	**94**
pc4	**-92**	**-57**	**79**	**79**
pc3	**-98**	**-75**	**84**	**77**
kc1	**-97**	**-81**	**96**	**87**
pc2	**-61**	**-90**	**33**	**89**
insurance	**-100**	**-98**	**97**	**97**
glass	**-71**	-5	**86**	**40**
ecoli	**-62**	**-56**	**83**	**80**
balance	**-77**	**-32**	**75**	4
car	**-43**	14	**65**	-5

Table 3.13 reports the correlations of G-mean and AUC with respect to Q-statistic and GD. Even though Table 3.11 shows reduced overall accuracy as diversity increases, both

G-mean and AUC present strong negative correlations with Q and positive correlations with GD in most cases. It means that increasing diversity leads to better G-mean and AUC. Therefore, we can say that diversity is beneficial to the overall performance, which better balances the performance between classes. The results answer the question of the relationship between ensemble diversity and overall performance in the class imbalance learning context, where a positive effect of diversity is found.

Additional Remarks

Comparing Table 3.5 and Table 3.12, we notice some differences of diversity's impact between artificial and real-world data sets. In practice, the positive effect of diversity seems to get weaker on overall accuracy and F-measure. More complex data distributions and feature correlations could be the reasons. As diversity helps to find more minority class examples, the majority class could suffer from great accuracy reduction.

Regarding the question of whether diversity is beneficial to the classification of the minority/majority classes in the presence of imbalanced data, the answer depends on the performance measures. Diversity improves minority-class recall on all artificial and real-world data sets, and minority-class F-measure on all artificial data sets and some real-world data sets. Diversity is generally harmful to minority-class precision. An opposite effect of diversity happens to the majority class.

The findings in this section are quite meaningful. Even if several authors found very little empirical relationship between overall accuracy and diversity of classification ensembles (Kuncheva and Whitaker, 2003; Garcia-Pedrajas et al., 2005), we obtain strong correlations of single-class performance, G-mean and AUC with respect to two diversity measures. It may suggest that different types of classification problems should be considered for better understanding and exploiting the role of ensemble diversity in the future.

3.5 Chapter Summary

This chapter studies the relationships between ensemble diversity and performance measures for class imbalance learning in depth, aiming at the following questions: what is the impact of diversity on the classification performance? Does diversity have a positive effect on the classification of minority/majority class? We choose Q-statistic as the main diversity measure and consider three single-class performance measures including recall, precision and F-measure. The relationship to overall performance is also discussed empirically by examining G-mean and AUC for a complete understanding.

To answer the first question, we derive mathematical links between Q-statistic and the single-class measures. This part of work is based on Kuncheva et al.'s pattern analysis (Kuncheva et al., 2003). We extend it to the single-class context under specific classification patterns of ensembles and explain why we expect diversity to have different impacts on minority and majority classes in class imbalance scenarios. Six possible behaving situations of these measures with respective to Q-statistic are obtained. As the second contribution of this chapter, we verify the measure behaviours empirically on a set of artificial and real-world imbalanced data sets. We examine the impact of diversity on each type of classes through correlation analysis. Strong correlations between diversity and classification performance are found, in which diversity is measured by Q-statistic and GD. We show the positive effect of diversity in recognizing minority class examples and balancing the trade-off between recall and precision of the minority class. It degrades the classification performance of the majority class in terms of recall and F-measure on real-world data sets. Diversity is beneficial to the overall performance evaluated by G-mean and AUC.

Significant and consistent correlations and the positive effect of diversity found in this chapter encourage us to develop novel ensemble learning algorithms for class imbalance learning in the following chapters that can make best use of our diversity analysis here, so that the importance of the minority class can be better considered.

CHAPTER 4

NEGATIVE CORRELATION LEARNING FOR CLASSIFICATION ENSEMBLES

This chapter proposes a new ensemble algorithm, named “AdaBoost.NC”, by combining ideas of negative correlation learning (NCL) (Liu and Yao, 1999b,a; Liu et al., 2000) and Boosting (Freund and Schapire, 1997). It aims at an effective, flexible and efficient NCL algorithm for classification problems. Existing NCL algorithms perform well in generalization by considering diversity explicitly during the training process, but suffer from some known drawbacks as we explained in section 2.4. To address their issues and extend to classification, two questions need to be answered in the algorithm design: how to express diversity and how to introduce diversity into training in the classification context. Section 4.2 derives an ambiguity measure based on the 0-1 error function for the first question. Section 4.3 proposes the AdaBoost.NC algorithm incorporating the ambiguity measure, which answers the second question. Experimental evaluation of its generalization performance and parameter setting is included. Section 4.4 provides more theoretical and empirical evidences to show its effectiveness. Section 4.5 summarizes the chapter. This chapter focuses on general classification problems.

4.1 Introduction

Since diversity was recognized as a main reason for the success of an ensemble algorithm, a number of researchers are encouraged to develop negative correlation learning (NCL) algorithms and work on related theoretical studies. Aiming at better generalization, NCL is an ensemble learning technique that takes into account ensemble diversity explicitly during training (Liu and Yao, 1999a), so as to achieve a better balance between individual performance and error correlation among the ensemble members. A typical NCL algorithm introduces a penalty term containing error correlation information into the error function of each individual learner, which is usually a neural network, so that they can be trained interactively.

However, the theoretical support of NCL is only valid in the regression context. There is no single theoretical framework or agreed definition of diversity for classification (Brown, 2010). It is still a challenging problem of considering diversity in the training of classification ensembles. Besides, existing NCL algorithms, such as CELS (Liu and Yao, 1999b), NCCD (Chan and Kasabov, 2005a) and RNCL (Chen and Yao, 2009), require base learners that are capable of producing posterior probabilities. So far, only neural networks have been applied in the literature as we have shown in section 2.4. Its computational cost can be high for large data sets and large ensembles. A more flexible and efficient NCL algorithm is desirable.

To overcome the above problems and obtain a flexible and efficient NCL algorithm for classification tasks with good generalization ability maintained, this chapter proposes an ensemble algorithm, called AdaBoost.NC, which incorporates an ambiguity term with the diversity information of the classification ensemble into the training framework of AdaBoost (Freund and Schapire, 1996). Different from the one derived from the quadratic error of regression ensembles and used in other NCL algorithms (Krogh and Vedelsby, 1995), the ambiguity term in AdaBoost.NC is decomposed from the 0-1 error function (Chen, 2008). The effectiveness and efficiency of AdaBoost.NC are analyzed thoroughly on some general classification data sets in this chapter, including the parameter setting issue. Its generalization is further studied in terms of error correlation by providing theoretical and empirical evidences. If and how it can benefit class imbalance problems will be discussed in the following chapters.

4.2 An Ambiguity Measure for Classification Ensembles

In the standard NCL training paradigm that incorporates a penalty term to encourage diversity in the procedure of minimizing the quadratic error function (Liu and Yao, 1999b), this penalty is derived from the ambiguity decomposition for regression ensembles (Krogh and Vedelsby, 1995):

$$\left(\bar{f} - y\right)^2 = \sum_i \alpha_i \left(f_i - y\right)^2 - \sum_i \alpha_i \left(f_i - \bar{f}\right)^2, \tag{4.1}$$

where y is the target of a data input, α_i is the weight of the i-th learner f_i with the constraint $\sum_i \alpha_i = 1$, and $\bar{f}$ is the linear combination of individuals $\bar{f} = \sum_i \alpha_i f_i$. The quadratic error of the ensemble is composed of two terms. The first term is the weighted average of individual errors. The second term is the ambiguity, expressing the prediction variability within the ensemble. In other words, ambiguity can be viewed as the difference between average individual error and ensemble error, i.e. *ambiguity = average individual error - ensemble error*. With this understanding, the idea was applied to the 0-1 error function for binary classification problems in (Chen, 2008). We extend it to general cases here based on the oracle output.

Given a data set $Z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ with c possible class labels $Y = \{\omega_1, \omega_2, \ldots, \omega_c\}$, we assume that $f_i(x_j)$ is the actual output of the i-th classifier for any data input x_j. We define the output of the ensemble with L components as

$$\bar{f}(x_j) = \arg\max_{\hat{y} \in Y} \sum_{i=1}^{L} \alpha_i \left[f_i(x_j) = \hat{y}\right], \tag{4.2}$$

where $\sum_{i=1}^{L}\alpha_i = 1$. $[\pi]$ is equal to 1 if π holds and 0 otherwise. Without loss of generality, we assume that $\sum_{i=1}^{L}\alpha_i\left[f_i\left(x_j\right)=y_j\right] \geq \sum_{i=1}^{L}\alpha_i\left[f_i\left(x_j\right)\neq y_j\right]$ is a necessary condition for the ensemble to produce a correct label. When the context is clear, 'x_j' will be omitted in the following equations for brevity of notation. Based on the definition of the 0-1 loss function, the error of a classifier f can be formulated as

$$error\left(f\right) = 1 - \left[f = y\right], \tag{4.3}$$

where $f = f_i, \bar{f}\left(i = 1, 2, \ldots, L\right)$ and y is the expected output. The difference between the average individual error and ensemble error is

$$\sum_{i=1}^{L}\alpha_i error\left(f_i\right) - error\left(\bar{f}\right) = \sum_{i=1}^{L}\alpha_i\left(\left[\bar{f} = y\right] - \left[f_i = y\right]\right). \tag{4.4}$$

The obtained term on the right-hand side of Eq. 4.4 is defined as the ambiguity for the 0-1 error function and denoted by *amb* in the following discussions. It measures the disagreement degree among individual classifiers in terms of correct and incorrect votes. Two ambiguity terms in Eq. 4.1 and Eq. 4.4 present a similar effect on generalization performance – a large classification ambiguity with individual performance maintained ensures a low ensemble error. The difference between them is that, the ambiguity in Eq. 4.4 can be negative, depending on whether the combined output of the ensemble is correct. If $\bar{f}$ is correct, *amb* will be positive, varying in range $[0, 1/2]$; otherwise, it will be negative, varying in range $[-1/2, 0]$. If all classifiers produce correct or incorrect votes (no disagreement), *amb* will be 0.

Ambiguity and Margin

Schapire et al. (Schapire et al., 1998) introduced the concept of margin associated with examples for the combined classifier. It is defined as the difference between the sum of the weights of individual classifiers assigned to the correct label and the maximum sum of the weights assigned to any single incorrect label. If we define a ψ function as

$$\psi\left(f\right)=\begin{cases}1, & \text{if } f=y\\ -1, & \text{if } f\neq y\end{cases} \tag{4.5}$$

then for a binary classification task, the margin of the ensemble on example x can be expressed as

$$m\left(x\right)=\sum_{i=1}^{L}\alpha_i\psi\left(f_i\left(x\right)\right). \tag{4.6}$$

It is a number in range $[-1,1]$. An example is correctly labelled if and only if its margin is positive. It measures the "confidence" of the classifications. Schapire et al. proved that a larger margin on the training data set guarantees an improvement in the upper bound on the generalization error of the ensemble. Both Bagging and Boosting methods have been shown to increase the margins and converge to a margin distribution in which most examples have large margins (Schapire et al., 1998). Especially, Boosting is aggressive in this effect.

Following the definition of ψ, the *amb* term can be reformulated as

$$amb=\frac{1}{2}\sum_{i=1}^{L}\alpha_i\left(\psi\left(\bar{f}\right)-\psi\left(f_i\right)\right), \tag{4.7}$$

which is equivalent to Eq. 4.4. The equation $\psi\left(\bar{f}\right)=sign\left(m\right)$ holds, indicating if the ensemble produces the correct label. Therefore, we obtain the relationship between the margin and ambiguity:

$$amb=\frac{1}{2}\left(sign\left(m\right)-m\right). \tag{4.8}$$

It is not hard to understand this equation. The margin expresses the classification confidence, while the ambiguity measures how ambiguous the classification is. When $sign\left(m\right)$ is unchanged, the more confidence, the less ambiguity.

Ambiguity and Entropy

We find a connection between *amb* and the diversity measure of entropy (*Ent*) (Cunningham and Carney, 2000) introduced in section 2.3.1, when the individual classifiers are uniformly weighted. Let $l(x)$ be the number of correct votes from the individuals on any input x (i.e. $\sum_{i=1}^{L}[f_i = y]$). The entropy on x is

$$Ent = \frac{\min\{l, L-l\}}{L - \lceil L/2 \rceil}. \tag{4.9}$$

According to Eq. 4.9 and Eq. 4.4,

$$amb = \frac{L \cdot [\bar{f} = y] - l}{L} = \begin{cases} \frac{Ent(L-\lceil L/2\rceil)}{L} & \text{if } \bar{f} \text{ is correct} \\ \frac{-Ent(L-\lceil L/2\rceil)}{L} & \text{if } \bar{f} \text{ is wrong} \end{cases} \tag{4.10}$$

In other words,

$$|amb| = \frac{Ent\,(L - \lceil L/2 \rceil)}{L}. \tag{4.11}$$

$|amb|$ and *Ent* are discriminated by a constant. The magnitude of *amb* expresses the absolute difference among the ensemble members. Again, its sign indicates whether an example is correctly classified by the ensemble.

4.3 A New NCL Algorithm – AdaBoost.NC

To overcome the problems of poor flexibility and high computational cost of exiting NCL algorithms, a new ensemble learning algorithm, AdaBoost.NC, is proposed by exploiting the training framework of AdaBoost (Freund and Schapire, 1996). It is regarded as a NCL algorithm for the reason that the *amb* term is introduced into training to encourage ensemble diversity, which is the essential idea of NCL. In this section, we first describe the algorithm in detail, including the explanations of the effect of *amb* in the algorithm and why it is expected to be effective and efficient. Then, its performance is investigated through extensive experiments on some general benchmark classification problems. Finally, we discuss how the accuracy of AdaBoost.NC gets affected by its parameters.

4.3.1 The AdaBoost.NC Algorithm

The key point of AdaBoost.NC is to emphasize the training examples that cause the ensemble to present low voting disagreement among the individual classifiers during learning, in addition to the examples misclassified by the current classifier. This is accomplished by applying a penalty term involving *amb* in the sequential training procedure of AdaBoost. The classification difference measured by *amb* for each training example is combined into its weight at each iteration. The weight-updating rule of AdaBoost is modified, such that both classification errors and low diversity will be penalized by rising weights. Table 4.1 presents the pseudo-code of AdaBoost.NC following the AdaBoost.M1 procedure (Freund and Schapire, 1996).

In step 3 of the algorithm, a penalty term p_i is calculated for each training example at the i-th iteration, in which the magnitude of amb_i assesses the "pure" disagreement degree within current ensemble $\bar{f}_i$ composed of the existing i classifiers. Uniform weights $(\frac{1}{i})$ are simply assigned to the individuals for the calculation, i.e.

$$amb_i = \frac{1}{i} \sum_{t=1}^{i} \left([\bar{f}_i = y] - [f_t = y] \right). \tag{4.12}$$

p_i is introduced into the weight-updating step (step 5). The main effect of applying p_i is that, the misclassified examples by the current classifier that receive more same votes from existing individual classifiers will get a larger weight increase; the correctly classified examples that get more different votes will get a larger weight decrease. Thus, both accuracy and diversity are considered during training.

The pre-defined parameter λ controls the strength of applying p_i. The choice of α_i in step 4 is decided by using the same inferring method in (Schapire and Singer, 1999; Sun et al., 2007) to bound the overall training error. From the form of α_i, we can see

Table 4.1: The AdaBoost.NC algorithm (Wang et al., 2010).

```
Given training data set $\{(x_1, y_1), \ldots, (x_j, y_j), \ldots, (x_N, y_N)\}$
with labels $y_j \in Y = \{\omega_1, \ldots, \omega_c\}$ and penalty strength $\lambda$,
initialize data weights $D_1(x_j) = 1/N$; penalty term $p_1(x_j) = 1$.

For each training epoch $i = 1, 2, \ldots, L$:
Step 1. Train weak classifier $f_i$ using distribution $D_i$.
Step 2. Get weak classifier $f_i$: $X \rightarrow Y$.
Step 3. Calculate the penalty value for every example $x_j$:
        $p_i(x_j) = 1 - |amb_i(x_j)|$.
Step 4. Calculate $f_i$'s weight $\alpha_i$ according to error and penalty:
        $\alpha_i = \frac{1}{2}\log\left(\frac{\sum_{j, y_j = f_i(x_j)} D_i(x_j)(p_i(x_j))^{\lambda}}{\sum_{j, y_j \neq f_i(x_j)} D_i(x_j)(p_i(x_j))^{\lambda}}\right)$
Step 5. Update data weights $D_i$ and obtain new weights $D_{i+1}$
        according to error and penalty:
        $D_{i+1}(x_j) = \frac{(p_i(x_j))^{\lambda} D_i(x_j) exp(-\alpha_i [f_i(x_j) = y_j])}{Z_i}$,
        where $Z_i$ is a normalization factor.

Output the final ensemble:
$\bar{f}(x) = \arg\max_{\hat{y} \in Y} \sum_{i=1}^{L} \alpha_i [f_i(x) = \hat{y}]$.
(Define $[\pi]$ to be 1 if $\pi$ holds and 0 otherwise.)
```

that the classifier having fewer misclassified examples, which receive larger classification ambiguity from the current ensemble members, will obtain a higher weight. It agrees with the common understanding about diversity in classification that we would not want ensemble members to make the same wrong decision. The details of how to choose α_i are given in the Appendix. B section.

Here are some additional points to explain the choice of the penalty term. Why do we use $|amb|$ to encourage diversity instead of using *amb*? As we have explained above, the sign of *amb* indicates whether an example is correctly classified by the ensemble. This accuracy information is already reflected in the original weight-updating rule of AdaBoost. There is no point to consider it repeatedly in p_i. For the same reason, the individual classifier is uniformly weighted for the calculation of *amb* instead of using α terms. The α_i's are already included in the exponential term of the weight-updating rule in step 5. In addition, we would not know f_i's weight α_i until step 4 when calculating p_i in step 3.

AdaBoost.NC can be viewed as the first NCL algorithm developed specifically for classification problems. Its training strategy is much simpler than existing NCL algorithms. It is free of choosing any base learning methods, whereas others are restricted to neural networks. The error correlation information is introduced into the weights of training examples, rather than the error function of the learners such as in the CELS algorithm (Liu and Yao, 1999b) or the training examples themselves such as in the NCCD algorithm (Chan and Kasabov, 2005a). In this way, diversity is considered from the ensemble level without modifying the base learner that makes the algorithm learner-dependent and training examples that can cause undesirable noise. All these features help to create a flexible ensemble training framework with improved efficiency and accuracy.

Finally, it is worth explaining why AdaBoost.NC is possible to outperform the conventional AdaBoost. Although AdaBoost attempts to enforce classifiers to make different errors by focusing on misclassified examples sequentially, the overfitting problem has been reported empirically (Quinlan, 1996; Opitz and Maclin, 1999; Dietterich, 2000a), especially when the processing data is noisy. In some cases, the weight vectors can become very skewed, which may lead to undesirable bias towards some limited groups of data. It is also found that AdaBoost can produce a diverse ensemble at the first few training epochs, but diversity drops as more classifiers are built (Shipp and Kuncheva, 2002b). So, it is suggested to stop the training progress early for performance enhancement with proper diversity maintained.

From a theoretical point of view, Schapire et al. derived an upper bound on the generalization error of AdaBoost (Schapire et al., 1998). They proved that AdaBoost is aggressive at increasing margins of the training examples, which contributes to the reduction of generalization error even after the training error reaches zero. However, this bound is rather weak (Schapire, 2002). Breiman found that a better margin distribution does not lead to a lower test error necessarily. If they tried too hard to make the margins larger, then overfitting set in (Breiman, 1999). Murua presented an improved error bound for the linear classifier combination by introducing “mutual weak dependence” (Murua, 2002). As an important progress, it is shown that both the low dependence between classifiers and large margins play an important role in achieving low error rates, and there is a trade-off between them. It provides the evidence that only considering margins is not sufficient. It is worth looking for a training procedure that can keep the dependence between the classifiers low with large margins. AdaBoost.NC works for this purpose. It is expected to alleviate the overfitting problem of AdaBoost, because the penalty term boosts not only the most difficult misclassified examples, but also the easiest examples that have been correctly labelled. Easy examples are more likely to be chosen to help the classification on difficult examples in AdaBoost.NC than in AdaBoost. It reduces the chance of focusing on the same group of misclassified examples. Both theoretical and empirical studies have provided justifications for AdaBoost.NC to achieve good performance.

4.3.2 Experimental Studies

In this section, we compare AdaBoost.NC to the conventional AdaBoost, CELS (Liu and Yao, 1999b) and NCCD (Chan and Kasabov, 2005a). CELS and NCCD are two typical NCL algorithms with different means to encourage diversity, which have been introduced in section 2.4.

Experimental Design

In our experiments, we choose MLP neural networks (NN) and C4.5 decision trees as the base learner. CELS and NCCD are designed for neural networks. It is necessary to discuss neural network ensembles. Each network is trained by 250 epochs. The updating epoch of NCCD is set to 20, which is a rough estimate, because the optimal update interval is problem-dependent. The decision tree is recognized as the “ideal” learner with AdaBoost, since its accuracy can be boosted dramatically. Therefore, we also examine the performance of AdaBoost.NC using C4.5 base learners.

Regarding the parameter λ in NCL algorithms that controls the strength of encouraging diversity, it is required to lie inside the range $[0, 1]$ in CELS and $[0, 1)$ in NCCD. In our experiments, we set it to $\{0.25, 0.5, 1\}$ for CELS and $\{0.5, 0.8\}$ for NCCD. Both of them use the "winner-take-all" strategy to combine the individual outputs as demanded by the original papers. The range of λ in AdaBoost.NC is not theoretically bounded. Based on some preliminary experiments showing that the performance of AdaBoost.NC is not very sensitive to the change of λ, we vary λ within the range [0.25(conservatively), 12(aggressively)]. The best result produced by each of these algorithms joins the following comparisons. Every ensemble consists of 9 individual classifiers, where setting an odd number is to avoid even voting. More parameter discussions for AdaBoost.NC will be included in the next section.

The experiments are conducted on a collection of UCI classification tasks (Frank and Asuncion, 2010), including 10 two-class data sets and 4 multi-class data sets. Some data sets are provided with a separate test set. For the others, we randomly sample 80% of the data used for training. The rest are for the test. The data information is summarized in Table 4.2, ordered by the training set size. Reported results are the averages over 10 runs of each algorithm. Student T-test with 95% confidence level is performed for significance analysis.

Performance Analysis

We output the test error rate and computation time produced by AdaBoost.NC (abbr. NC), AdaBoost (abbr. Ada), CELS and NCCD in Tables 4.3 - 4.6. The computation environment is windows Xp with Intel Core 2 Duo 2.13GHz and 1G RAM. All the algorithms are implemented in Java.

(1) Generalization error

Tables 4.3 and 4.4 present the results on the two-class benchmarks for MLP and C4.5

Table 4.2: Summary of benchmark data sets ('d' denotes a discrete attribute; 'c' denotes a continuous attribute).

Data	Train	Test	Attributes	Classes
promoter	84	22	57d	2
sonar	166	42	60c	2
ionosphere	280	71	34c	2
house-votes-84	348	87	15d	2
crx	489	201	9d+6c	2
breast-w	559	140	9d	2
pima	614	154	8c	2
german	800	200	13d+7c	2
hypothyroid	2530	633	18d+7c	2
insurance	5822	4000	85c	2
soybean-large	307	376	35d	19
vowel	528	462	10c	11
segmentation	1848	462	19c	7
satimage	4435	2000	36c	6

ensembles respectively. The lowest error rate among the four methods is in boldface. If there is a statistically significant difference between AdaBoost.NC and one of the other algorithms, the latter will be marked with a '*'.

When MLP is the base learner, AdaBoost.NC gives lower error rates than AdaBoost on the 10 data sets consistently, 5 of which are significant. When comparing to CELS, AdaBoost.NC produces better generalization performance on 5 out of 10 data sets, where 3 wins are significant; CELS wins twice significantly and the other three are ties. When comparing to NCCD, AdaBoost.NC wins on 7 data sets significantly, and the remaining three are ties. NCCD does not perform well among the four methods. The reason could be that it adjusts training data directly, which may induce noise and lead to very inaccurate classifiers. When C4.5 is the base learner, similar to the results of MLP, AdaBoost.NC outperforms AdaBoost in the last 6 data sets significantly, and the remaining 4 cases are ties.

For these two-class problems, generally speaking, AdaBoost.NC presents very promising generalization performance. It improves AdaBoost and outperforms NCCD signifi-

Table 4.3: "Mean ± standard deviation" of test error rate (abbr. Err %) and mean computation time (in seconds) of AdaBoost.NC (abbr. NC), AdaBoost (abbr. Ada), CELS and NCCD on two-class data sets with MLP as the base learner. The lowest error rate for each data set is in boldface. The significant difference between AdaBoost.NC and one of the other algorithms is indicated by '*'.

Data		NC	Ada	CELS	NCCD
promoter	Err	17.727±4.520	24.091±4.312*	**12.727±1.917***	15.000±9.103
	Time	19.6	19.5	1666.4	267.3
sonar	Err	10.952±2.300	12.381±3.214	**7.619±1.004***	22.857±15.681*
	Time	43.2	43.1	267.1	271.4
ionosphere	Err	4.647±0.950	5.211±0.950	4.366±0.445	**4.084±2.523**
	Time	26.5	26.5	174.6	25.0
house-votes	Err	**2.528±0.484**	3.218±0.726*	3.333±0.363*	4.023±1.817*
	Time	9.9	10.0	72.3	9.8
crx	Err	**15.671±0.853**	19.055±1.502*	16.667±0.586*	18.358±1.454*
	Time	12.1	12.0	426.9	69.7
breast-w	Err	**1.071±0.505**	1.286±0.451	3.214±1.398*	33.071±28.081*
	Time	7.2	7.1	1939.6	298.5
pima	Err	24.025±1.960	25.130±1.437	**23.246±1.396**	49.221±17.987*
	Time	7.5	7.4	60.1	8.4
german	Err	**24.900±1.776**	26.900±1.744*	25.250±1.585	31.850±6.200*
	Time	31.7	31.7	1351.2	209.1
hypothyroid	Err	**2.227±0.157**	2.275±0.260	2.432±0.388	2.796±0.372*
	Time	137.7	137.6	957.6	185.0
insurance	Err	6.450±0.177	8.080±0.171*	**6.293±0.228**	6.407±0.271
	Time	2874.4	2852.7	17930.0	3742.2

cantly on more than half of the data sets. AdaBoost.NC and CELS are quite competitive with each other.

We further compare their performance on the multi-class data sets. The results are shown in Tables 4.5 and 4.6. AdaBoost.NC shows slightly better performance than AdaBoost. Except the soybean-large data set where NCCD wins the other three methods, AdaBoost.NC outperforms CELS and NCCD significantly on data sets vowel and segmentation. Besides, we notice that CELS does not show any advantages over these multi-class data sets. In fact, its ability to solve multi-class problems has not been fully investigated in the literature to date. It would be interesting to compare AdaBoost.NC and CELS with existing multi-class approaches in future work.

In summary, performance improvements are observed in both types of learners by using AdaBoost.NC, especially on the two-class data sets. Because AdaBoost.NC enforces

Table 4.4: "Mean ± standard deviation" of test error rate (abbr. Err %) and mean computation time (in seconds) of AdaBoost.NC (abbr. NC) and AdaBoost (abbr. Ada) on two-class data sets with C4.5 as the base learner. The lower error rate for each data set is in boldface. The significant difference between AdaBoost.NC and AdaBoost is indicated by '*'.

Data		NC	Ada
promoter	Err	**10.454±4.815**	10.909±6.843
	Time	0.014	0.01
sonar	Err	21.667±5.318	**21.190±5.435**
	Time	0.212	0.206
ionosphere	Err	**1.831±0.680**	2.253±0.984
	Time	0.225	0.205
house-votes	Err	3.793±1.217	**3.563±1.006**
	Time	0.059	0.051
crx	Err	**16.069±1.050**	18.109±0.943*
	Time	0.111	0.127
breast-w	Err	**1.357±0.710**	2.143±0.588*
	Time	0.035	0.028
pima	Err	**20.649±1.779**	23.961±2.340*
	Time	0.20	0.25
german	Err	**25.400±1.926**	27.600±2.389*
	Time	0.21	0.26
hypothyroid	Err	**0.553±0.083**	0.948±0.235*
	Time	0.81	1.43
insurance	Err	**6.225±0.079**	8.120±0.225*
	Time	52.3	52.5

misclassified examples receiving a low disagreement within the current ensemble to get the largest weight increase and the correctly classified examples receiving a high disagreement to get the largest weight decrease, both of the "more difficult" and "easier" examples are emphasized through the penalty term. We infer that the superiority of AdaBoost.NC is attributed to the focus of "easier" examples that facilitates the classification on "more difficult" examples, instead of only boosting the misclassified ones.

(2) Computation time

According to Tables 4.3 - 4.6, AdaBoost.NC is the winner in training time among all the NCL algorithms with no doubt. Computing the penalty term does not bring much extra computational cost compared to AdaBoost on these data sets. It saves a large amount of training effort compared to CELS and NCCD. The running time of

Table 4.5: "Mean ± standard deviation" of test error rate (abbr. Err %) and mean computation time (in seconds) of AdaBoost.NC (abbr. NC), AdaBoost (abbr. Ada), CELS and NCCD on multi-class data sets with MLP as the base learner. The lowest error rate for each data set is in boldface. The significant difference between AdaBoost.NC and one of the other algorithms is indicated by '*'.

Data		NC	Ada	CELS	NCCD
soybean-large	Err	10.080±1.124	10.665±1.281	10.505±0.258	**7.952±0.714***
	Time	71.2	68.8	1251.5	226.3
vowel	Err	**44.826±1.834**	46.147±1.679	47.792±2.456*	49.329±1.846*
	Time	23.2	23.1	150.1	45.7
segmentation	Err	2.575±0.387	**2.489±0.371**	3.463±0.478*	3.376±0.423*
	Time	107.8	107.8	717.8	374.2
satimage	Err	**9.735±0.382**	10.175±0.408*	9.850±0.309	9.785±0.233
	Time	607.4	606.5	3702.5	2225.4

Table 4.6: "Mean ± standard deviation" of test error rate (abbr. Err %) and mean computation time (in seconds) of AdaBoost.NC (abbr. NC) and AdaBoost (abbr. Ada) on multi-class data sets with C4.5 as the base learner. The lower error rate for each data set is in boldface. The significant difference between AdaBoost.NC and AdaBoost is indicated by '*'.

Data		NC	Ada
soybean-large	Err	**7.553±1.457**	8.244±1.795
	Time	0.17	0.15
vowel	Err	**51.428±2.217**	53.030±2.105
	Time	0.40	0.38
segmentation	Err	**2.099±0.323**	2.316±0.250
	Time	1.3	1.2
satimage	Err	**10.315±0.395**	10.700±0.297*
	Time	9.7	9.6

AdaBoost.NC is primarily determined by the ensemble size and the size of the given classification task, which is the same as every other ensemble algorithm. From Table 4.3, in which the data sets are in the order of the training set size, we observe that the training time of AdaBoost.NC increases along with the increasing numbers of training examples and attributes of the data set. For example, "insurance" has the longest training time due to its largest data size. "Ionosphere" has shorter training time than "sonar" because it has fewer data attributes. The same happens to CELS and NCCD. Differently, their computation time further depends on the type of attributes. CELS and NCCD take much longer time in training if the data set has more discrete attributes. Every possible value of a discrete attribute corresponds to an input node in the network whereas each continuous attribute only creates one in our current algorithm implementation. Data with more discrete attributes thus makes the NN have a larger structure and contain more training parameters. It increases the efforts of calculating penalty for the diversity and updating weights greatly. In AdaBoost.NC, the penalty calculation is independent of the structure of the base learner. Therefore, AdaBoost.NC seems robust to whether the attribute is discrete or real-valued. For example, the training time of CELS and NCCD on "breast-w" is longer than on "pima" for all discrete attributes of "breast-w", which is not observed in AdaBoost.NC. Because any base learning algorithm is applicable to AdaBoost.NC, the training time can be further reduced by using simple learners. It would be useful to do complexity analysis theoretically here in the future.

Based on the results and analysis, we conclude that AdaBoost.NC inherits the advantages of NCL methods with good generalization performance. Meanwhile, it has a more flexible training framework and requires much less training time.

4.3.3 Parameter Discussions

In this section, we investigate how AdaBoost.NC's performance gets affected by two predefined parameters involved in the algorithm, i.e. penalty strength λ and ensemble size L. We observe the changing tendency of test error by varying one parameter and fixing the other shown in plots. AdaBoost is compared as the baseline method. The analysis here provides guidance for choosing appropriate parameter values for AdaBoost.NC.

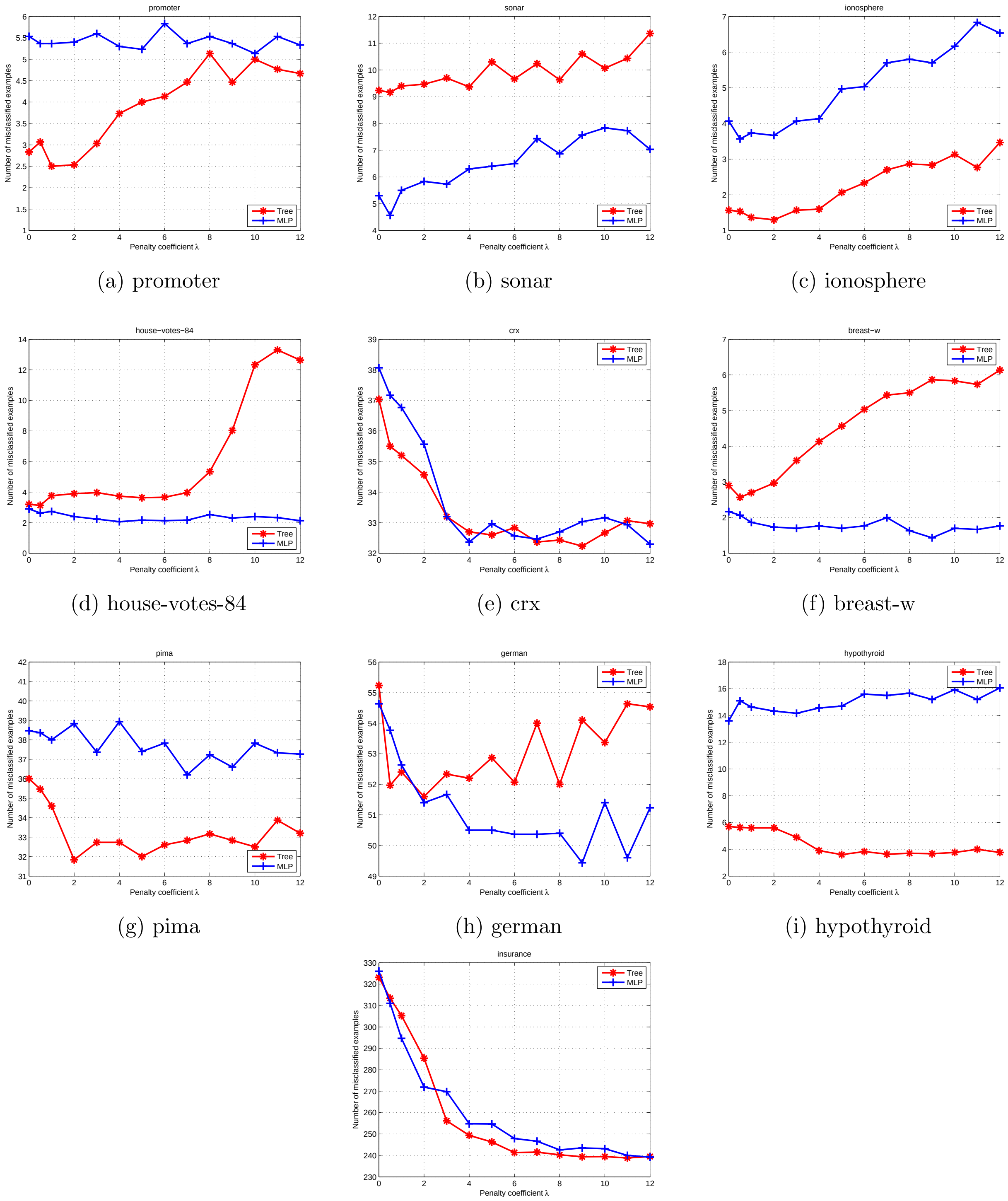
The experiments are conducted on the ten two-class data sets in Table 4.2. As before, we randomly sample 80% of the data as the training set and use the rest for testing, if no testing file is provided. All results in the following graphs are averages over 30 runs of the algorithm for each parameter setting.

Penalty Strength λ

To observe the effect of λ, we fix $L = 9$ and vary λ in the range of $[0, 12]$. When λ equals to 0, AdaBoost.NC is reduced to AdaBoost. Fig. 4.1 presents the changing tendency of test error along with λ for each data set. The first point on the curve indicates the number of errors made by AdaBoost. Each plot includes two curves: one is produced by MLP ensembles; the other is produced by C4.5 tree ensembles.

According to Fig. 4.1, we have the following observations: 1) AdaBoost.NC with a proper λ can generally improve the classification performance in both tree and MLP cases, especially on the last six data sets with larger sizes (Fig. 4.1(e)-(j)). However, the best λ is domain-dependent. For example, the best performance is achieved when λ is approximately 4 for "crx"; the error keeps decreasing until λ reaches 12 for "insurance". 2) The best λ is also related to the base learner. In the plot of "german" (Fig. 4.1(h)), the lowest error is achieved at $\lambda = 2$ on the tree curve, but $\lambda = 9$ on the MLP curve. 3) There is no evidence to show which base learner works consistently better with AdaBoost.NC. The tree ensemble seems more sensitive to the change of λ. It is more likely to suffer from a performance reduction by increasing λ, such as in "promoter", "house-votes-84" and "breast-w". 4) In most cases, tree-based AdaBoost.NC presents a fast drop of classification error when λ is small than 4. Therefore, $\lambda \leq 4$ is deemed to be a conservative range of setting λ. As λ becomes larger, there could be either a further performance improvement (such as in "insurance") or a performance degradation (such as in "german").

Generally speaking, AdaBoost.NC can achieve better generalization than AdaBoost by choosing an appropriate λ. However, the optimal λ depends on problem domains and base learners. The tree ensemble seems more sensitive to λ than the MLP ensemble. Only an approximate range of λ is obtained here based on how it affects the effectiveness of AdaBoost.NC. Some remaining issues are: when would a large λ cause a performance degradation and how large would it be? A theoretical range of λ would be very useful for wider application of the algorithm. It will be discussed as possible future work.



(a) promoter (b) sonar (c) ionosphere

(d) house-votes-84 (e) crx (f) breast-w

(g) pima (h) german (i) hypothyroid

(j) insurance

Figure 4.1: Performance behaviour of AdaBoost.NC by varying λ. Ensemble size L is set to 9. (X-axis: penalty strength $\lambda \in [0, 12]$; Y-axis: number of misclassified examples.)

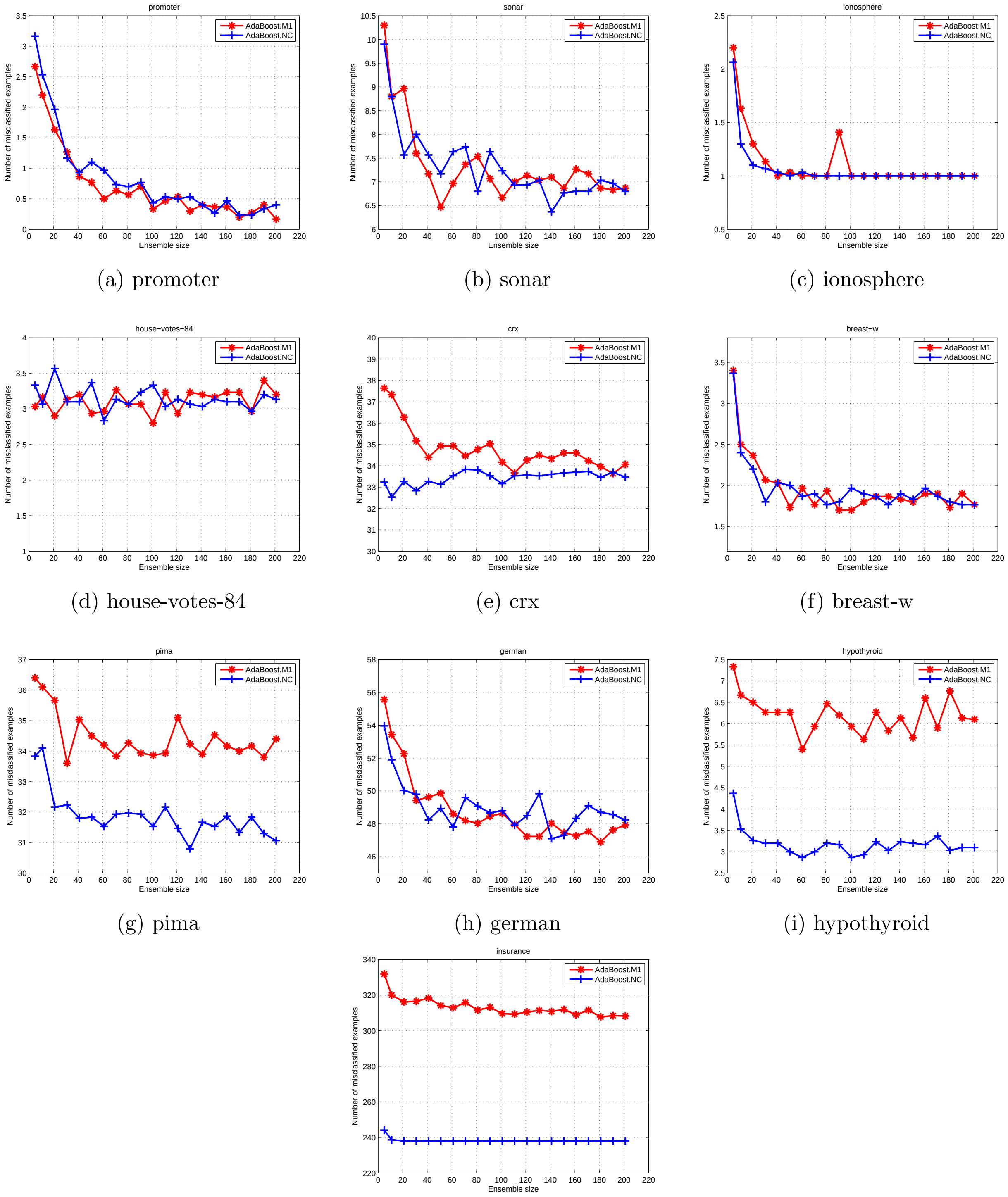
Ensemble Size L

Now, let's see the changing tendency of test error by varying ensemble size L from 5 to 201. Each plot in Fig. 4.2 depicts two curves produced by AdaBoost and AdaBoost.NC respectively with tree base learners. λ in AdaBoost.NC is set to the optimal value obtained in Fig. 4.1.

According to Fig. 4.2, we have the following observations: 1) the performance of both AdaBoost and AdaBoost.NC is not reduced by increasing L. No overfitting phenomenon is observed. 2) The performance of both AdaBoost and AdaBoost.NC becomes stable when L gets approximately 41-51. A fast drop of error usually happens when L is smaller than 41. The error reduction slows down when L is larger than 51. 3) AdaBoost.NC seems to be more effective than AdaBoost when L is relatively small ($L \leq 21$ in general), such as in "ionosphere", "crx", "breast-w" and "german". 4) As L becomes larger, AdaBoost.NC and AdaBoost tend to reach a comparable performance level in some cases, such as "crx", "breast-w" and "german". In some other cases, AdaBoost.NC outperforms AdaBoost consistently, which is not affected by L, such as in "pima", "hypothyroid" and "insurance". The underlying reason needs a further investigation. Generally speaking, increasing the ensemble size does not reduce the performance of AdaBoost.NC, which becomes stable as more classifiers are built. It is more effective when the size is relatively small.

4.4 AdaBoost.NC and Error Correlation

Since AdaBoost.NC is claimed to be a "negative correlation" learning algorithm, it should be able to reduce the error correlation within the ensemble. We look into the AdaBoost.NC algorithm in terms of error correlation from both theoretical and experimental aspects in this section. We will prove how the error correlation among classifiers relates to classification accuracy, and show if AdaBoost.NC can reduce this correlation with improved performance.



(a) promoter (b) sonar (c) ionosphere

(d) house-votes-84 (e) crx (f) breast-w

(g) pima (h) german (i) hypothyroid

(j) insurance

Figure 4.2: Performance behaviour of AdaBoost and AdaBoost.NC by varying L. Penalty strength λ is set to the optimal value obtained in Fig. 4.1. (X-axis: ensemble size $L \in [5, 201]$; Y-axis: number of misclassified examples.)

Error Correlation and Classification Accuracy

Before discussions, we need to define error correlation precisely. Given any data point (x, y) with a noise-free target, the *degree of error correlatedness* for any pair of classifiers f_i and f_k can be defined as (Ali and Pazzani, 1995)

$$cov\left(f_i \neq y, f_k \neq y\right) = p\left(f_i \neq y, f_k \neq y\right) - p\left(f_i \neq y\right) p\left(f_k \neq y\right). \tag{4.13}$$

$p\left(f_i \neq y, f_k \neq y\right)$ denotes the probability that f_i and f_k make errors simultaneously. $p\left(f_i \neq y\right) p\left(f_k \neq y\right)$ is the likelihood that would be obtained if they make errors in a statistically independent manner. It is not hard to get $cov\left(f_i \neq y, f_k \neq y\right) = cov\left(f_i = y, f_k = y\right)$. If $cov\left(f_i = y, f_k = y\right)$ is positive (negative), then the classifiers make errors in a positively (negatively) correlated way. A value of 0 indicates that they are uncorrelated.

A more important question here is: *how is "cov" related to classification accuracy?* To simplify our problem, we focus our attention on the sequential training context of AdaBoost with the majority voting combination for binary classification. In addition, we make the assumption that any individual classifier f_i at the i-th training iteration is only correlated with its previous one f_{i-1} in making errors, considering that the weights of examples for selecting training examples for the next classifier in AdaBoost are updated only based on the accuracy of the current classifier without taking other existing ones into account. In other words, for any $t < i - 1$, we have $cov\left(f_i = y, f_t = y\right) = 0$. With these conditions, Theorem 4.1 is obtained to describe how the accuracy of ensemble $\bar{f}_i$ composed of the first i classifiers is related to the error correlation between classifier f_i and ensemble $\bar{f}_{i-1}$ composed of the first $(i-1)$ classifiers $(i > 2)$.

Theorem 4.1. *In the sequential training context with the majority voting combination, reducing the error correlation between f_i and $\bar{f}_{i-1}$ can improve the classification accuracy of $\bar{f}_i$ under the assumption of $cov\left(f_i = y, f_t = y\right) = 0\,(t < i - 1)$.*

Proof. For clear presentation, we establish some notations first. Imagine that we are building the i-th classifier f_i currently. To make the first i classifiers produce a correct combined result based on majority vote, we assume that they need to provide at least n correct votes. We define:

- $\bar{f}_i$ represents the combination of the first i classifiers; "$\bar{f}_i = y^n$" means that these i classifiers contain at least n correct votes.
- $\bar{f}_{i-1}$ represents the combination of the first $(i-1)$ classifiers; "$\bar{f}_{i-1} = y^n$" means that these $(i-1)$ classifiers contain at least n correct votes.
- $\bar{f}^{n-1}_{i-1}$ represents any possible combination of $(n-1)$ classifiers randomly chosen from the first $(i-1)$ classifiers; "$\bar{f}^{n-1}_{i-1} = y^{n-1}$" means that all the chosen classifiers give the correct label, i.e. $(n-1)$ correct votes.

With these notations, the accuracy of $\bar{f}_i$ can be expressed as

$$p\left(\bar{f}_i = y^n\right) = p\left(\bar{f}_{i-1} = y^n\right) + \sum_{t=1}^{\binom{i-1}{n-1}} \left[p\left(f_i = y, \bar{f}^{n-1}_{i-1(t)} = y^{n-1}\right) - p\left(f_i = y, \bar{f}^{n-1}_{i-1(t)} = y^{n-1}, \bar{f}_{i-1} = y^n\right)\right], \tag{4.14}$$

Eq. 4.14 considers majority voting accuracy in two separate situations. To have $\bar{f}_i$ make a correct decision, one possibility is that the first $(i-1)$ classifiers already contain at least n correct votes (the first term on the right-hand side). The other probability is that the first $(i-1)$ classifiers produce exactly $(n-1)$ correct votes and the i-th classifier f_i is correct (the terms in the summation).

Now we focus on the expression inside the summation that involves f_i. According to the definition of *cov*, we obtain

$$\begin{aligned} p & \left(f_i = y, \bar{f}^{n-1}_{i-1} = y^{n-1}\right) - p\left(f_i = y, \bar{f}^{n-1}_{i-1} = y^{n-1}, \bar{f}_{i-1} = y^n\right) \\ = & \; \Omega + cov\left(f_i = y, \bar{f}^{n-1}_{i-1} = y^{n-1}\right) - cov\left(f_i = y; \bar{f}^{n-1}_{i-1} = y^{n-1}, \bar{f}_{i-1} = y^n\right), \end{aligned} \tag{4.15}$$

where $\Omega = p\left(f_i = y\right)\left[p\left(\bar{f}^{n-1}_{i-1} = y^{n-1}\right) - p\left(\bar{f}^{n-1}_{i-1} = y^{n-1}, \bar{f}_{i-1} = y^n\right)\right]$. By applying the above assumption of error correlation in sequential training, $\bar{f}^{n-1}_{i-1}$ in the *cov* terms is reduced to f_{i-1}, if f_{i-1} is contained in $\bar{f}_{i-1}^{n-1}$. Otherwise, $cov\left(f_i = y, \bar{f}_{i-1}^{n-1} = y^{n-1}\right) = 0$. Thus, Eq. 4.15 can be reformulated into two possible cases:

- If $f_{i-1} \notin \bar{f}_{i-1}^{n-1}$,

$$\Omega - cov\left(f_i = y, \bar{f}_{i-1} = y^n\right). \tag{4.16}$$

- If $f_{i-1} \in \bar{f}_{i-1}^{n-1}$,

$$\Omega + cov\left(f_i = y, f_{i-1} = y\right) - cov\left(f_i = y; f_{i-1} = y, \bar{f}_{i-1} = y^n\right). \tag{4.17}$$

Due to the positive property of term $\left[p\left(\bar{f}_{i-1}^{n-1} = y^{n-1}\right) - p\left(\bar{f}_{i-1}^{n-1} = y^{n-1}, \bar{f}_{i-1} = y^n\right)\right]$, $p\left(f_i = y\right)$ should be kept as high as possible. Meanwhile, $p\left(\bar{f}_i = y^n\right)$ is negatively related to $cov\left(f_i = y, \bar{f}_{i-1} = y^n\right)$ and $cov\left(f_i = y; f_{i-1} = y, \bar{f}_{i-1} = y^n\right)$. It implies that reducing the error correlation between f_i and $\bar{f}_{i-1}$ can improve the ensemble accuracy of $\bar{f}_i$. In other words, it makes f_i play a better role in the ensemble. This proof also suggests a trade-off between individual accuracy and error correlation. □

It can be imagined that the training examples receiving all correct/wrong votes from the first $(i-1)$ classifiers tend to cause higher $cov\left(f_i = y, \bar{f}_{i-1} = y^n\right)$ than those that get partially correct/wrong votes. According to the training strategy of AdaBoost.NC, it boosts the group of data with larger *cov* through the penalty term. Meanwhile, the accuracy of each classifier is not overlooked, because the examples misclassified by the current classifier will get the attention back immediately in the next round through the exponential term inside the weight-updating rule. Each classifier in AdaBoost.NC is enforced to balance individual accuracy and error correlation sequentially.

Empirical Evidence of Error Correlation for AdaBoost.NC

To verify our theoretical result and further illustrate how AdaBoost.NC works, we track its error correlation and test error during training on a two-class artificial data set in comparison with the conventional AdaBoost. The artificial data set is generated from two Gaussian distributions with equal covariance and a small overlapping area close to the separating line. Each class contains 200 training examples. A separate testing data set is produced by using the same generation method and settings, in which each class has 50 points.

In the experiments, we build 51 C4.5 decision trees to form an ensemble. Penalty strength λ is set to 0, 2, and 9 respectively. When $\lambda = 0$, AdaBoost.NC is reduced to AdaBoost. Each setting is repeated 30 times. At each iteration of building a new classifier, we output the average $cov\left(f_i = y, \bar{f}_{i-1} = y^n\right)$ value and error rate on testing data in Fig. 4.3, where x-axis presents the training epoch increasing from 2 to 51 and y-axis indicates the correlation and test error rate respectively.

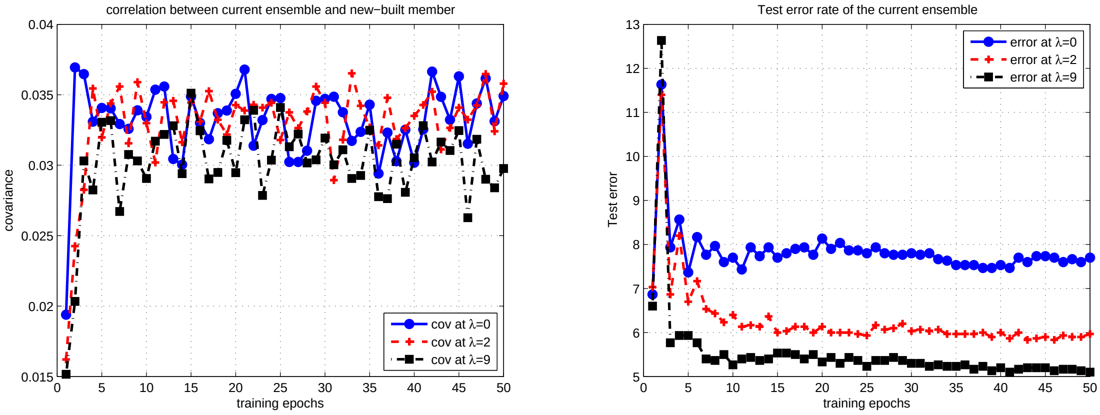
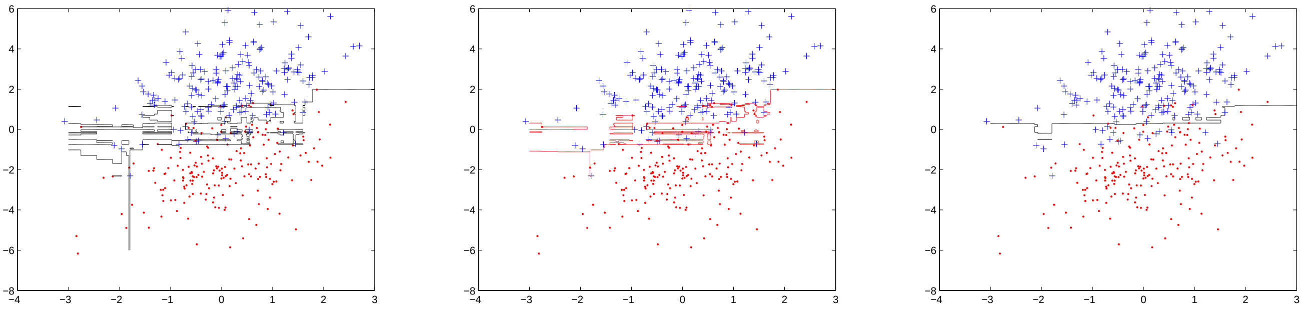


Figure 4.3: Error correlation and test error rate (in %) during sequential training.

According to Fig. 4.3, the correlation values on the $\lambda = 9$ curve are generally smaller than values at $\lambda = 0$, especially during the first few training iterations. Correspondingly, AdaBoost.NC with $\lambda = 9$ produces the lowest error rate among the three settings. This observation agrees with Theorem 4.1 and confirms the ability of AdaBoost.NC to reduce error correlation.

Ali and Pazzani mentioned an error rearrangement procedure to explain how negatively correlated errors can be beneficial to an ensemble's performance (Ali and Pazzani, 1995). Basically, their analysis claimed that, if a classifier has to make an error, then make the error on a different example. Sorting errors in a negatively dependent way can minimize the number of ensemble errors. The training mechanism of AdaBoost.NC can be viewed as an automatic error rearrangement procedure. The least ambiguous and correctly classified training examples are boosted, to reduce the chance for classifiers to make same errors. So, we believe that AdaBoost.NC encourages both classification accuracy and negatively correlated errors.

Next, we illustrate the classification boundaries produced by AdaBoost.NC on the same artificial data, in order to understand how the performance improvement happens graphically. Fig. 4.4 presents three classification boundary plots produced by AdaBoost.NC with λ set to $\{0, 2, 9\}$ respectively.



(a) $\lambda = 0$ (b) $\lambda = 2$ (c) $\lambda = 9$

Figure 4.4: Classification boundaries of AdaBoost.NC with $\lambda \in \{0, 2, 9\}$.

As λ gets larger, AdaBoost.NC produces smoother boundaries and less overfits the overlapping region that is supposed to contain the most difficult data points. When $\lambda = 0$, the boundary includes many small squares and spiky regions, because the algorithm keeps boosting examples from these areas. Although they are more likely to be selected to join the next training session, they are still hard to separate on their own. When $\lambda = 2$, the situation is relieved to some extent. Fewer small squares are produced. When $\lambda = 9$, more "easier" examples are selected to help the training of each classifier, which results in less overfitting boundaries further and thus the best generalization performance.

In summary, the analysis in this section tells us that AdaBoost.NC is capable of reducing error correlation during the sequential training. The effect is reflected in a lower error rate and less overfitting classification boundaries.

4.5 Chapter Summary

In this chapter, we propose a NCL algorithm for classification ensembles, called "AdaBoost.NC", in order to overcome the problems of poor flexibility and high computational cost of existing NCL algorithms. It combines the strength of NCL and Boosting. First, an ambiguity term is derived from the 0-1 error function, which describes the classification difference within the ensemble and provides theoretical support to adapt the traditional NCL based on the squared error function in regression for classification. Then, the ambiguity term containing diversity information is introduced into the training framework of AdaBoost, where the weight-updating rule is modified, so that both accuracy and diversity are encouraged during the sequential training procedure.

The effectiveness and efficiency of AdaBoost.NC are evaluated through extensive experiments in comparison with two typical NCL algorithms (i.e. CELS and NCCD) and the conventional AdaBoost on a group of benchmark classification problems. It achieves better accuracy than NCCD and AdaBoost, and shows competitive results with CELS. It uses much less training time than both CELS and NCCD. Besides, any base learners can be applied to AdaBoost.NC.

Next, we examine the performance behaviour of AdaBoost.NC along with two predefined parameters, i.e. penalty strength λ and ensemble size L. The optimal λ depends on problem domains and base learners. Tree-based AdaBoost.NC presents higher sensitivity to a large λ than the one using neural networks. AdaBoost.NC's performance is not reduced by a large L. It is more effective when L is comparatively small.

The effectiveness of AdaBoost.NC is further investigated with more theoretical and empirical evidences provided in terms of error correlation. We prove that reducing the error correlation between the currently-built classifier and the ensemble composed of the already-existing ones sequentially can improve final classification accuracy of the whole ensemble. AdaBoost.NC shows the ability to reduce this error correlation and achieves a lower test error than AdaBoost by providing less overfitting classification boundaries.

Generally speaking, AdaBoost.NC is a promising ensemble algorithm that inherits the good generalization ability of NCL and the flexible training strategy of AdaBoost. This chapter only discussed some general classification tasks without considering data distributions. How it performs in class imbalance problems will be investigated in the following chapters.

CHAPTER 5

NEGATIVE CORRELATION LEARNING FOR TWO-CLASS IMBALANCE PROBLEMS

Many ensemble methods have been proposed to deal with class imbalance problems. Their common strategy is to rebalance the training subsets or make base classifiers cost-sensitive. However, finding a proper parameter setting is always a crucial and problem-dependent issue, such as the resampling rate and an optimal cost matrix, without which overfitting and over-generalization are likely to occur. Therefore, it is desirable to develop an alternative method to address "imbalance". In chapter 3, we showed the positive effect of ensemble diversity in solving class imbalance problems with theoretical and empirical evidences, which suggested a potential solution. Following on from this part of work, we proposed a NCL algorithm called AdaBoost.NC for classification problems in chapter 4. It encourages ensemble diversity explicitly during training and shows good generalization performance in general classification problems. We are thus motivated to find out *if and how we can take advantage of ensemble diversity to better deal with class imbalance* in this chapter. An associated question is, *if NCL methods including AdaBoost.NC can be good solutions to the class imbalance problem.*

To answer these questions, section 5.2 investigates the generalization ability of AdaBoost.NC in depth under some artificial imbalanced scenarios and discusses the factors that may affect its effectiveness. Section 5.3 compares it with other NCL algorithms and state-of-art ensemble methods in class imbalance learning through comprehensive experiments on a set of real-world data sets with highly skewed class distributions. Finally, section 5.4 concludes the chapter. This chapter focuses on two-class imbalance problems.

5.1 Introduction

Since many standard machine learning algorithms were found to suffer from classification difficulties when tackling class imbalance problems, a number of data-level and algorithm-level algorithms have been proposed to help to recognize minority class examples accurately and achieved varying degrees of success in generalization. In particular, ensemble learning has drawn growing attention in this field. Some ensemble solutions have been widely discussed with empirical success, such as BEV (Li, 2007), SMOTEBoost (Chawla et al., 2003), JOUS-Boost (Mease et al., 2007), RareBoost (Joshi et al., 2001), etc. They are often integrated with data-level and cost-sensitive strategies, in which individual classifiers are adapted for emphasizing the minority class, and combining multiple classifiers are used to lower down the risk of overfitting and reduce the error variance. However, data-level strategies manipulate training data directly, and thus need extra care in choosing proper parameters before use. Both overfitting and over-generalization issues have been reported in oversampling and undersampling techniques (Chawla et al., 2002; Mease et al., 2007; He and Garcia, 2009). Besides, it is always a problem of deciding which resampling method to use with ensembles, depending on the given task and performance evaluation method. Cost-sensitive ensemble methods demand explicit cost information of classes prior to learning, which is hard to obtain in many real-world situations. Therefore, we would like to explore an alternative method to overcome these problems. Meanwhile, it should have good generalization performance especially in the minority class.

In our earlier work in chapter 3, we studied the role of ensemble diversity in the classification of class imbalance problems. A positive effect of diversity was found, which inspires the idea of using ensemble diversity to solve our problem. It leads our focus to NCL (Liu and Yao, 1999b), an effective ensemble learning technique that explicitly manages the accuracy-diversity trade-off during training. In chapter 4, we proposed a new ensemble algorithm AdaBoost.NC using the idea of NCL for classification tasks. In addition to its simplicity and efficiency, it shows very promising generalization performance. Taking this step further, we study how NCL algorithms perform in class imbalance problems including AdaBoost.NC in this chapter. The potential advantage of using NCL is that it improves generalization without giving up any data information from the majority class or generating new data points for the minority class. Hence, inappropriate choices of resampling techniques and sampling rates can be avoided. Besides, all NCL-related works only focus on the effectiveness in terms of overall accuracy to our best knowledge. It is obviously not appropriate for imbalance learning scenarios. Instead, AUC and single-class performance measures are examined as another interesting point in this chapter.

NCL is investigated on both artificial and real-world imbalanced data sets with comprehensive experimental discussions. Three NCL methods (i.e. AdaBoost.NC (Wang et al., 2010), CELS (Liu and Yao, 1999b), NCCD (Chan and Kasabov, 2005a)) and three training strategies (i.e. learning from the original imbalanced data, random oversampling, random undersampling) are integrated and compared. The results suggest that the combination of the random oversampling strategy and NCL can improve the prediction for the minority class without losing the overall performance compared to other class imbalance learning methods. The aim of applying oversampling is simply to maintain sufficient number of minority class examples for the NCL algorithm to increase diversity. Moreover, we show that AdaBoost.NC performs the best with decision tree base learners, which presents high AUC and minority-class measures by producing broader and less overfitting classification boundaries for the minority class. This chapter concentrates on two-class imbalance problems, where only one majority class and one minority class exist in the data set. Multi-class cases will be included in the next chapter.

5.2 AdaBoost.NC for Two-Class Imbalance Problems

Before we move on to our main work, it is worth explaining why we believe that NCL might be helpful for class imbalance learning. A major reason for the difficulty in identifying rare examples is that the classification boundaries are too tight around them. The majority class takes over the data space. In other words, ignoring or overfitting the minority class is very likely to happen to each individual classifier in an ensemble. The individuals tend to make the same errors on the minority class and thus a low diversity degree within the ensemble can be imagined. The unbalanced distribution diminishes the effect of ensemble techniques (Chawla and Sylvester, 2007). Generated classification rules can be very specific and useless for the future prediction (He and Garcia, 2009; Chawla et al., 2002). To overcome this problem, existing ensemble algorithms adjust class distributions by either applying different over/under-sampling techniques to every ensemble member or assigning higher costs to rare examples during training. The boundaries are thus expanded towards the majority class space (Monard and Batista, 2002; Chawla et al., 2002). From a different view of the ensemble level, a reasonable conjecture would be that diversity was introduced into the classification of the minority class or both classes indirectly by applying those techniques, since the same errors are less likely to occur.

To make the ensemble less overfit training data and achieve better generalization performance, one way is to increase diversity. Overfitting is associated with the error variance in terms of the bias-variance decomposition of the generalization error (Sun et al., 2007; Cunningham, 2000). If overfitting happens, predictions will vary between examples. Encouraging ensemble diversity can reduce this variance (Brown et al., 2005).

Since NCL promotes diversity explicitly among ensemble members, it is believed to alleviate the classification difficulty of class imbalance problems to some extent. Using NCL could be an implicit way of emphasizing the minority class. Besides, Chawla and Sylvester have noticed that the classification performance on imbalanced data sets can be improved by exploiting ensemble diversity in making different types of errors, which can be viewed as one of the earliest studies on this issue (Chawla and Sylvester, 2007).

In this section, we examine the effectiveness of AdaBoost.NC in identifying rare cases in depth on several two-dimensional artificial data sets with different imbalance rates. We focus on the algorithm itself and investigate the effect of penalty term that controls diversity. Decision boundaries are presented to illustrate if and how it can improve the classification performance. Considering that it is also a variation of Boosting, AdaBoost is compared as the baseline method in our analysis. Both single-class and overall performance measures are discussed. In order to show the advantages of using AdaBoost.NC to handle class imbalance problems, we perform an effect analysis of parameters in AdaBoost.NC through the ANOVA experiment (short for "ANalysis Of VAriance"). More comparisons with state-of-art ensemble methods will be given in the next section.

5.2.1 Artificial Data Sets and Experimental Settings

Two-class artificial data are generated from two Gaussian distributions with equal covariance and a small overlapping area close to the separating boundary. Three imbalanced degrees are discussed. Concretely, one class always keeps 200 training points, while the size of the other class is varied in {10 (very imbalanced), 50 (imbalanced), 200 (balanced)}. They are denoted as "200-10", "200-50" and "200-200". We also generate a testing file by using the same method, in which each class has 50 points.

In addition to AdaBoost.NC, a single classifier model and AdaBoost are built for the comparison. Every ensemble consists of 51 base classifiers, which is an appropriate size for an ensemble to produce stable performance based on the observation in section 4.3.3. An odd number is to avoid even voting. The C4.5 decision tree and MLP neural network (NN) are chosen as the base learner and reported in separate subsections. Each type is implemented in Weka (Witten and Frank, 2005), an open source data mining tool. The default Weka parameters are used to train C4.5. MLP (Ripley, 1996) is built with one hidden layer. The training epoch for MLP is set to 250. Other parameters use the default settings in Weka.

The penalty strength λ in AdaBoost.NC is set to 2 and 9. Based on the findings in section 4.3.3 (Wang and Yao, 2010), $\lambda = 2$ is a relatively conservative setting to show if AdaBoost.NC can make a performance improvement, and $\lambda = 9$ encourages diversity aggressively. For a full understanding of the feasibility of using ensemble diversity to facilitate class imbalance learning, both values are discussed here.

Three training strategies are applied to the learning methods before the training starts for imbalanced cases "200-50" and "200-10": learning from the original imbalanced data directly, random oversampling and random undersampling. The aim of considering resampling is simply to eliminate the negative effect of the imbalanced distribution and avoid the minority class being ignored during training. For oversampling, a new training set is formed by sampling from the minority class randomly with replacement until it reaches the same size as the majority class. Undersampling downsizes the majority class randomly without replacement until both classes are of equal size. Every experiment under each setting is repeated 30 times, and the averages are recorded. Student T-test with 95% confidence level is conducted for significance analysis.

In summary, the experiment considers two types of base learners, three training data sets and three training strategies. Four models are constructed for each combination of the above settings, including a single classifier, AdaBoost, AdaBoost.NC with $\lambda = 2$ (encourage diversity conservatively) and AdaBoost.NC with $\lambda = 9$ (encourage diversity aggressively). For brevity of notation, we use "NC" to denote AdaBoost.NC, "Ada" to denote the conventional AdaBoost and "Single" to denote a single classifier in the following analysis.

In regard to the evaluation criteria, we use AUC (Bradley, 1997) as the overall performance measure, and recall and precision to evaluate the performance for each class. AUC has proved to be robust to the class distribution. It is computed by altering a threshold value for labeling an example (Fawcett, 2003). Classification boundaries are presented graphically for a visual understanding of the obtained results.

5.2.2 AdaBoost.NC Tree Ensembles

Tables 5.1 - 5.2 present the measure outputs from tree ensembles on the three artificial data sets. The significantly best values among the four models are in boldface. Each entry presents the mean followed by the standard deviation.

Table 5.1: "Mean ± standard deviation" of AUC and single-class performance measures on "200-200" by tree-based models. The significantly best value is in boldface. (R-pos: recall of positive class; P-pos: precision of positive class; R-neg: recall of negative class; P-neg: precision of negative class.)

	200-200 Trees			
	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	0.950±0.000	0.978±0.006	0.979±0.003	**0.985±0.005**
R-pos	**0.940±0.000**	0.894±0.017	0.911±0.018	0.923±0.025
P-pos	0.959±0.000	0.951±0.009	**0.969±0.012**	**0.972±0.015**
R-neg	0.960±0.000	0.954±0.009	**0.970±0.012**	**0.973±0.015**
P-neg	**0.941±0.000**	0.900±0.014	0.916±0.016	0.927±0.021

For the balanced case "200-200" in Table 5.1, an arbitrary class is tagged as "positive". The other is treated as the "negative" class. "R-pos" and "P-pos" denote **r**ecall and **p**recision from the **pos**itive class respectively. "R-neg" and "P-neg" are those from the **neg**ative class. NC shows significantly better AUC than the single tree and AdaBoost. NC with $\lambda = 9$ makes a further improvement than NC with $\lambda = 2$. Within each class, NC attains higher recall and precision than Ada. The single tree is not bad at recall and precision to some extent, but the AUC measure is not satisfactory.

Now let's see the imbalanced cases. Table 5.2 organizes the measures into three groups according to the training strategy. "Origin" are the results produced by the models of learning from the imbalanced data without rebalancing. "Over" and "Under" groups are from those applying random oversampling or undersampling respectively before the training starts. "min" and "maj" are short for the minority and majority classes.

Among the four models, NC is capable of improving the overall performance measure AUC, which achieves the significantly highest values in all 6 cases when $\lambda = 9$. The single tree always gives the worst AUC.

Table 5.2: “Mean ± standard deviation” of AUC and single-class performance measures on “200-50” and “200-10” by tree-based models with three training strategies (Origin: original data; Over: oversampling; Under: undersampling). The significantly best value is in boldface. (R-min: recall of minority class; P-min: precision of minority class; R-maj: recall of majority class; P-maj: precision of majority class.)

200-50 Trees				
Origin	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	0.893±0.000	0.941±0.009	**0.956±0.010**	**0.960±0.011**
R-min	**0.840±0.000**	0.712±0.014	0.726±0.012	0.829±0.024
P-min	1.000±0.000	1.000±0.000	1.000±0.000	1.000±0.000
R-maj	1.000±0.000	1.000±0.000	1.000±0.000	1.000±0.000
P-maj	**0.862±0.000**	0.777±0.008	0.785±0.007	0.854±0.017
Over	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	0.888±0.030	0.941±0.009	0.949±0.010	**0.962±0.006**
R-min	0.777±0.043	0.710±0.013	0.735±0.012	**0.856±0.019**
P-min	1.000±0.000	1.000±0.000	1.000±0.000	0.996±0.011
R-maj	1.000±0.000	1.000±0.000	1.000±0.000	0.996±0.011
P-maj	0.819±0.029	0.775±0.008	0.790±0.007	**0.874±0.014**
Under	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	0.923±0.019	0.957±0.009	**0.964±0.010**	**0.963±0.011**
R-min	0.865±0.039	0.826±0.040	0.837±0.034	**0.882±0.019**
P-min	0.981±0.019	0.972±0.023	0.980±0.019	0.975±0.021
R-maj	0.982±0.018	0.976±0.021	0.983±0.017	0.977±0.019
P-maj	0.880±0.028	0.850±0.029	0.858±0.025	**0.893±0.013**

200-10 Trees				
Origin	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	0.890±0.000	0.899±0.014	0.910±0.012	**0.932±0.012**
R-min	0.460±0.000	**0.602±0.038**	**0.609±0.030**	0.421±0.195
P-min	**1.000±0.000**	0.991±0.014	**0.998±0.007**	0.966±0.182
R-maj	**1.000±0.000**	0.994±0.008	**0.999±0.005**	**1.000±0.000**
P-maj	0.649±0.000	**0.715±0.019**	**0.719±0.014**	0.643±0.085
Over	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	0.808±0.003	0.902±0.016	0.909±0.015	**0.927±0.013**
R-min	0.616±0.007	0.614±0.026	0.633±0.020	**0.785±0.025**
P-min	1.000±0.000	0.995±0.011	0.992±0.020	0.994±0.018
R-maj	1.000±0.000	0.997±0.006	0.995±0.013	0.995±0.015
P-maj	0.723±0.003	0.721±0.013	0.731±0.011	**0.823±0.017**
Under	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	0.911±0.030	**0.943±0.039**	**0.944±0.043**	**0.953±0.029**
R-min	**0.837±0.080**	**0.822±0.050**	**0.809±0.043**	0.747±0.102
P-min	0.985±0.020	0.994±0.014	0.986±0.068	0.999±0.007
R-maj	0.985±0.019	0.994±0.012	0.981±0.098	0.999±0.007
P-maj	**0.863±0.060**	**0.850±0.035**	**0.838±0.028**	0.802±0.053

When oversampling is applied, NC with $\lambda = 9$ produces the best minority-class recall for both data sets without losing any performance in the majority class. More minority class examples are found by setting a large λ, which shows the effect of having a high diversity level.

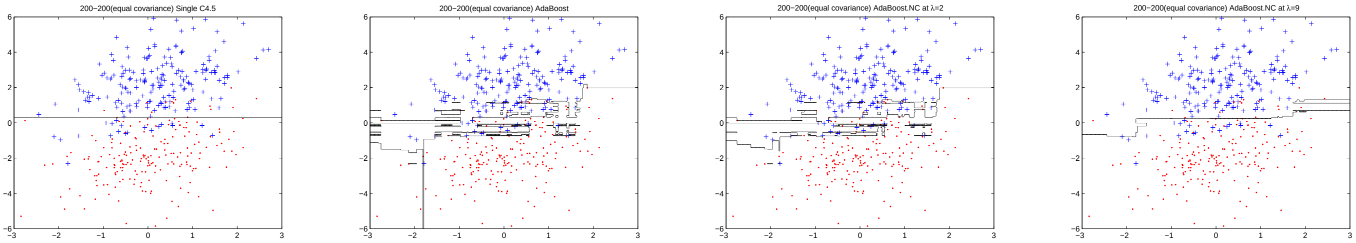
In the "Origin" group, NC does not show any consistent improvement. Especially for the very imbalanced case "200-10", the minority-class recall gets even lower by using a large λ. The minority-class precision and majority-class recall remain very high, most of which reach 1. It implies that NC does not improve the generalization of recognizing the minority class examples. It turns out that considering diversity alone without emphasizing the rare class in the NCL training is not enough. The imbalanced distribution weakens the effect of the penalty term in AdaBoost.NC, because the large number of majority class examples takes over the attention of the learning algorithm.

When undersampling is applied, the improvement brought by AdaBoost.NC is not as significant as that in the oversampling group. Our explanation is that undersampling itself has caused some classification diversity indirectly by removing some data information from the majority class. Increasing the diversity further with fewer training examples seems not to be helpful.

To sum up, AdaBoost.NC tree ensembles can improve AUC and find minority class examples effectively without sacrificing the performance on the majority class, when combined with the oversampling strategy. The algorithm itself is sensitive to the imbalance distribution. Undersampling and AdaBoost.NC seem not to work well together, due to the lost data information through undersampling.

To further understand the above observations and illustrate how AdaBoost.NC can be useful, corresponding boundary plots are presented in Figs. 5.1 - 5.7. For the case "200-200" in Fig. 5.1, AdaBoost apparently overfits the examples near the boundaries with some small regions. Boundaries produced by AdaBoost.NC are much simpler, which are smoothed further by increasing λ. AdaBoost.NC with $\lambda = 9$ least overfits the training examples. The boundary produced by the single tree is too simple to be as accurate as

AdaBoost.NC. Hence, we can say that AdaBoost.NC with $\lambda = 9$ yields the closest decision boundaries to the real one in this balanced scenario, which is consistent with the results in Table 5.1. The main role of the penalty term is to make the ensemble less overfit, such that better generalization performance is achieved.



(a) 200-200 single C4.5

(b) 200-200 AdaBoost

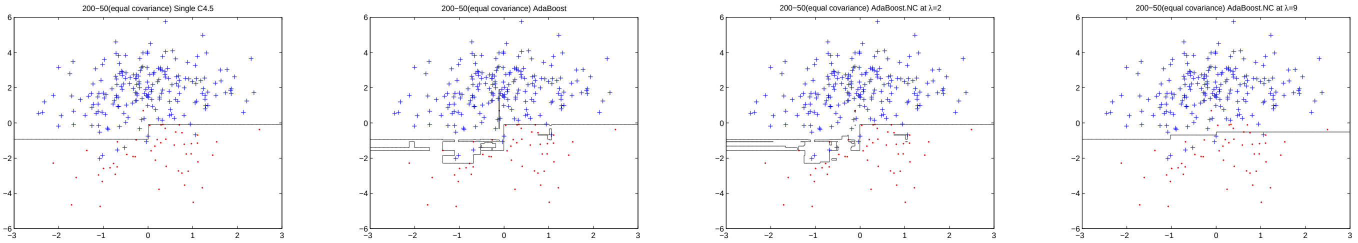
(c) 200-200 AdaBoost.NC with $\lambda = 2$

(d) 200-200 AdaBoost.NC with $\lambda = 9$

Figure 5.1: Classification boundaries on “200-200” by tree-based models.

For the two imbalanced data sets in Figs. 5.2 - 5.7, in addition to simpler boundaries, AdaBoost.NC produces broader space for the minority class towards the majority class side in “oversampling” plots (Fig. 5.3 and Fig. 5.6), which are closer to the real boundary. The effect of oversampling is to make the algorithm not overlook the classification diversity on the minority class. It is a meaningful result, since it breaks the overfitting problem of oversampling and makes use of the replicates of the rare examples to produce less overfitting boundaries. It is worth noting that this behaviour is not observed in the “original data” (Fig. 5.2 and Fig. 5.5) and “undersampling” (Fig. 5.4 and Fig. 5.7) plots. The boundaries may get even tighter.

The plots reveal that AdaBoost.NC tree ensembles are capable of improving both overall and minority-class performance measures for class imbalance problems, when used with random oversampling. It is achieved by generating less overfitting and broader classification boundaries for the minority class. In accordance with Table 5.2, it is not necessary to use AdaBoost.NC with undersampling, where the produced boundaries in the corresponding plots (Figs. 5.4 and 5.7) are quite simple and similar to each other.



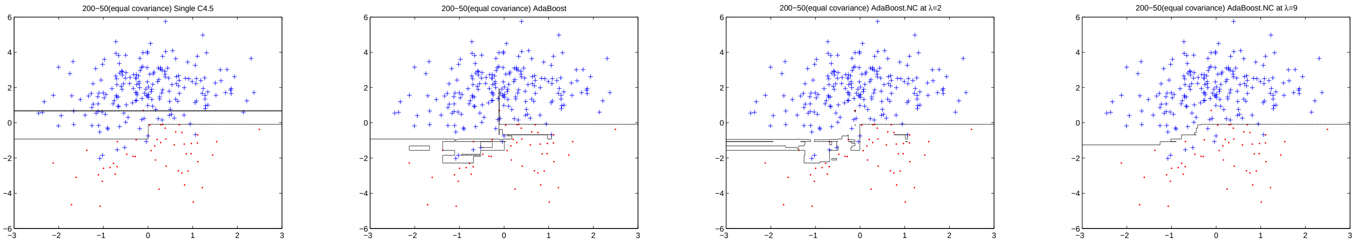
(a) 200-50 single C4.5

(b) 200-50 AdaBoost

(c) 200-50 AdaBoost.NC with $\lambda = 2$

(d) 200-50 AdaBoost.NC with $\lambda = 9$

Figure 5.2: Classification boundaries on "200-50" by tree-based models using original data.



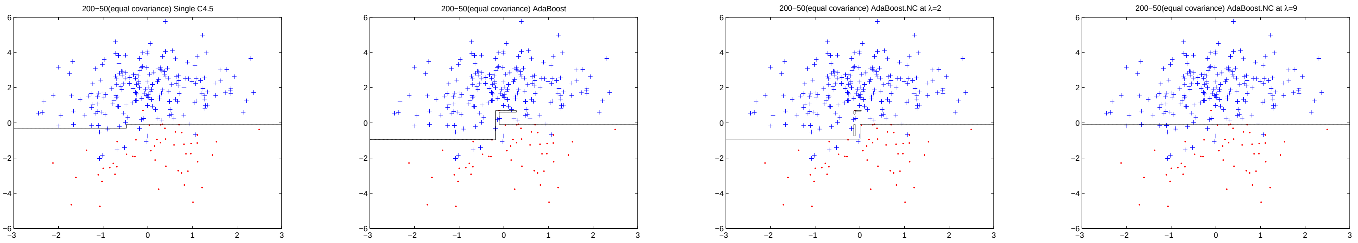
(a) 200-50 single C4.5

(b) 200-50 AdaBoost

(c) 200-50 AdaBoost.NC with $\lambda = 2$

(d) 200-50 AdaBoost.NC with $\lambda = 9$

Figure 5.3: Classification boundaries on "200-50" by oversampling + tree-based models.



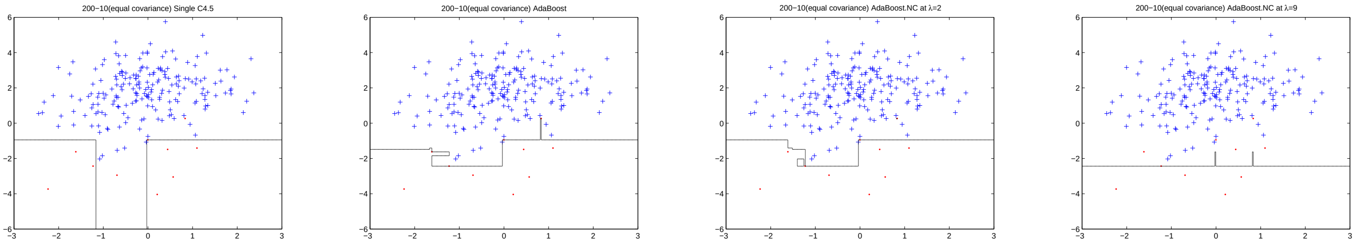
(a) 200-50 single C4.5

(b) 200-50 AdaBoost

(c) 200-50 AdaBoost.NC with $\lambda = 2$

(d) 200-50 AdaBoost.NC with $\lambda = 9$

Figure 5.4: Classification boundaries on "200-50" by undersampling + tree-based models.



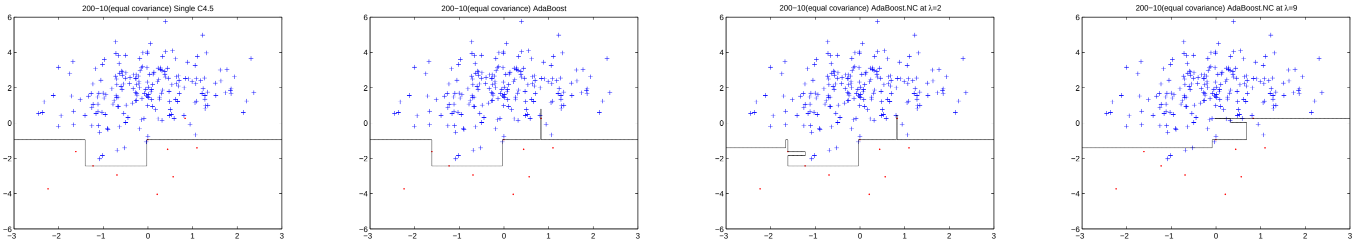
(a) 200-10 single C4.5

(b) 200-10 AdaBoost

(c) 200-10 AdaBoost.NC with $\lambda = 2$

(d) 200-10 AdaBoost.NC with $\lambda = 9$

Figure 5.5: Classification boundaries on "200-10" by tree-based models using original data.



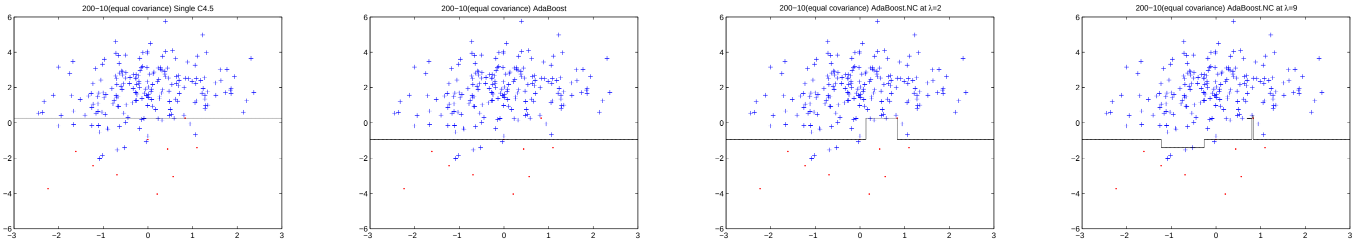
(a) 200-10 single C4.5

(b) 200-10 AdaBoost

(c) 200-10 AdaBoost.NC with $\lambda = 2$

(d) 200-10 AdaBoost.NC with $\lambda = 9$

Figure 5.6: Classification boundaries on "200-10" by oversampling + tree-based models.



(a) 200-10 single C4.5

(b) 200-10 AdaBoost

(c) 200-10 AdaBoost.NC with $\lambda = 2$

(d) 200-10 AdaBoost.NC with $\lambda = 9$

Figure 5.7: Classification boundaries on "200-10" by undersampling + tree-based models.

5.2.3 AdaBoost.NC Neural Network (NN) Ensembles

The same analysis is carried out for AdaBoost.NC NN ensembles. The experimental results are quite different from the ones produced by tree-based models. For the balanced data in Table 5.3, the single NN produces the best AUC. AdaBoost.NC does not bring any significant and consistent performance improvement. Fig. 5.8 plots the corresponding classification boundaries. The boundaries produced by the single NN and AdaBoost seem smoother than the ones produced by AdaBoost.NC.

Table 5.3: "Mean ± standard deviation" of AUC and single-class performance measures on "200-200" by NN-based models. The significantly best value is in boldface. (R-pos: recall of positive class; P-pos: precision of positive class; R-neg: recall of negative class; P-neg: precision of negative class.)

	200-200 NNs			
	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	**0.993±0.001**	0.984±0.004	0.990±0.002	0.988±0.004
R-pos	0.891±0.017	0.886±0.018	**0.937±0.034**	0.912±0.022
P-pos	**0.977±0.004**	**0.972±0.015**	0.928±0.033	0.954±0.023
R-neg	**0.979±0.005**	**0.974±0.015**	0.925±0.037	0.955±0.024
P-neg	0.900±0.015	0.896±0.015	**0.938±0.030**	0.916±0.018

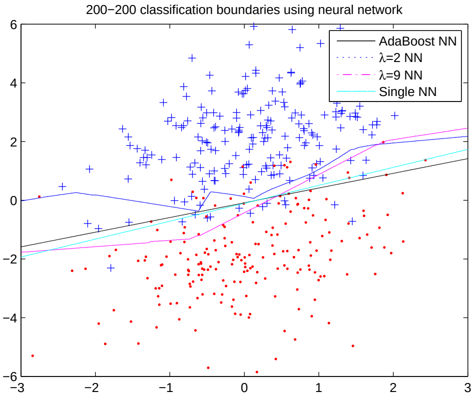


Figure 5.8: Classification boundaries on "200-200" by 4 NN-based models.

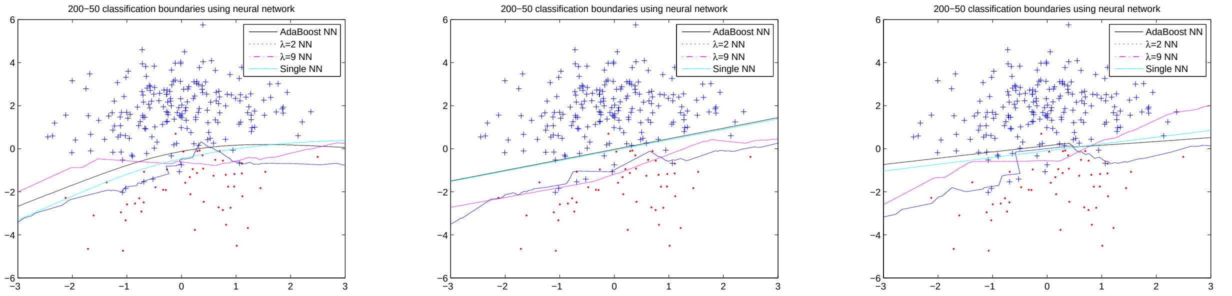
Performance results for the imbalanced cases are presented in Table 5.4. Unlike tree-based models where NC always gives the highest AUC, a single NN classifier attains the best AUC in all six settings under the imbalanced scenarios. On data set "200-10", the

Table 5.4: "Mean ± standard deviation" of AUC and single-class performance measures on "200-50" and "200-10" by NN-based models with three training strategies (Origin: original data; Over: oversampling; Under: undersampling). The significantly best value is in boldface. (R-min: recall of minority class; P-min: precision of minority class; R-maj: recall of majority class; P-maj: precision of majority class.)

200-50 NNs				
Origin	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	**0.991±0.000**	0.980±0.008	0.983±0.003	0.969±0.012
R-min	**0.861±0.008**	0.799±0.051	0.778±0.026	0.778±0.067
P-min	1.000±0.000	1.000±0.000	0.999±0.004	0.997±0.008
R-maj	1.000±0.000	1.000±0.000	0.999±0.003	0.997±0.006
P-maj	**0.878±0.006**	0.834±0.035	0.818±0.017	0.820±0.045
Over	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	**0.991±0.002**	0.965±0.012	0.983±0.005	0.976±0.012
R-min	**0.880±0.000**	**0.874±0.022**	0.798±0.032	0.790±0.081
P-min	0.986±0.010	0.994±0.009	**0.999±0.004**	0.984±0.016
R-maj	0.987±0.009	0.994±0.008	**0.999±0.003**	0.986±0.015
P-maj	**0.891±0.000**	**0.888±0.017**	0.832±0.022	0.828±0.049
Under	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	**0.993±0.001**	0.976±0.011	0.986±0.004	0.977±0.011
R-min	**0.891±0.016**	0.860±0.044	0.844±0.041	**0.898±0.045**
P-min	0.981±0.009	0.974±0.018	**0.988±0.011**	0.960±0.033
R-maj	0.982±0.008	0.976±0.017	**0.990±0.010**	0.961±0.036
P-maj	**0.900±0.013**	0.876±0.032	0.865±0.030	**0.906±0.036**

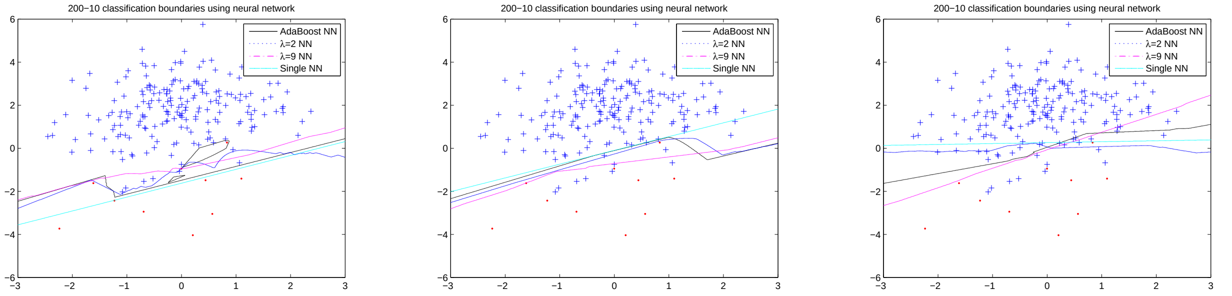
200-10 NNs				
Origin	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	**0.993±0.000**	0.969±0.012	0.956±0.008	0.947±0.006
R-min	0.631±0.023	0.685±0.028	0.758±0.030	**0.811±0.023**
P-min	**1.000±0.000**	**1.000±0.000**	**1.000±0.000**	0.996±0.008
R-maj	**1.000±0.000**	**1.000±0.000**	**1.000±0.000**	0.996±0.007
P-maj	0.731±0.013	0.761±0.016	0.806±0.019	**0.841±0.015**
Over	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	**0.992±0.003**	0.952±0.016	0.953±0.007	0.940±0.003
R-min	**0.880±0.000**	0.816±0.067	0.808±0.044	0.789±0.039
P-min	0.981±0.007	0.997±0.007	**1.000±0.000**	**0.999±0.004**
R-maj	0.982±0.006	0.997±0.006	**1.000±0.000**	**0.999±0.003**
P-maj	**0.891±0.000**	0.847±0.047	0.840±0.031	0.827±0.026
Under	Single	Ada	NC with $\lambda = 2$	NC with $\lambda = 9$
AUC	**0.991±0.003**	0.979±0.011	0.980±0.011	0.978±0.013
R-min	0.918±0.034	0.898±0.050	0.894±0.043	0.902±0.044
P-min	0.958±0.031	0.966±0.026	0.964±0.029	0.959±0.030
R-maj	0.958±0.036	0.967±0.027	0.965±0.033	0.960±0.031
P-maj	0.923±0.029	0.908±0.040	0.904±0.035	0.909±0.034

single NN suffers from some difficulty in finding minority class examples with low recall, but it is amended by the resampling strategies. The NC models, however, do not show a consistent improvement in any performance measure. Using resampling and varying λ are not very helpful. Figs. 5.9 - 5.10 further illustrate the results. It's hard to tell which boundary is consistently better. Generally speaking, "undersampling + a single NN" seems good enough to have good AUC and minority-class performance, where the NN guarantees a high AUC value and the undersampling helps to identify more minority class examples. The AdaBoost.NC NN ensemble does not show much benefit on these artificial data sets.



(a) NN models learning from original data (b) Oversampling + NN models (c) Undersampling + NN models

Figure 5.9: Classification boundaries on "200-50" by 4 NN-based models for three training strategies.



(a) NN models learning from original data (b) Oversampling + NN models (c) Undersampling + NN models

Figure 5.10: Classification boundaries on "200-10" by 4 NN-based models for three training strategies.

Comparing the measures between tree-based and NN-based models, although the tree-based NC with $\lambda = 9$ achieves the best AUC over the other tree-based models, it is still not as good as the NN-based ones. According to the boundary plots, the reason could be that the NN is able to produce a smoother and simpler classification boundary than the tree learner. This shows an advantage when the processing task is easy. The boundary generated by the NN is closer to the real one, and better AUC is achieved. However, the artificial data in this section only represent a limited class of imbalanced cases. Japkowicz has shown that the sensitivity of NN to the class imbalance increases with the complexity of the domain (Japkowicz, 2000b). More complex data distributions exist in real-world applications, which could be very noisy at the same time. Therefore, the performance between base learners will be further examined over a set of real-world problems in section 5.3.

5.2.4 The Analysis of Parameter Effects

The experimental results so far show that, to tackle class imbalance problems effectively, AdaBoost.NC requires the use of random oversampling, and a large λ value is preferable. In the current section, we provide an analysis of the impact of the oversampling rate and λ across different base learners and data sets. The analysis suggests that, even though AdaBoost.NC has two parameters to be tuned (oversampling rate and λ), it is robust to the choice of oversampling rate; the choice of λ does not depend on the imbalance rate of training data. So, the substantial advantage of our approach still holds.

We perform a mixed (split-plot) factorial analysis of variance (ANOVA) (Montgomery, 2004). The factors analyzed are the oversampling rate, λ, the base learner and the imbalance rate of training data. A mixed design is necessary because the oversampling rate, λ and the base learner are within-subject factors (their levels vary within a data set), whereas the imbalance rate is a between-subjects factor (its levels vary depending on the data set being used). The factorial design is a commonly adopted method for effect analysis, when there is more than one factor. It allows the effects of a factor to be estimated at several levels of the other factors. In most factorial experiments, two levels are considered for each factor. In our cases, artificial data sets "200-50" and "200-10" are used as training data with different imbalance rates. The oversampling rate is set to 100% and 150% respectively, which is the size ratio of the minority class to the majority class after oversampling is applied. λ is set to 2 (conservative level) and 9 (aggressive level) as before. The C4.5 decision tree and MLP neural network are chosen as the base learner. The effects of the factors on both AUC and minority-class recall are analyzed. These performance measures are referred to as responses in the context of ANOVA. AdaBoost.NC composed of 51 classifiers is repeated 30 times for each combination of factor level.

The ANOVA results are presented in Table 5.5, including the p-value and eta-squared (η^2). A p-value smaller than 0.05 indicates a significant difference by rejecting the null hypothesis under the significance level of 5%. η^2 is a measure in the range of $[0, 1]$ describing the effect size (Pierce et al., 2004). The larger the η^2, the greater the effect of the factor.

Table 5.5: ANOVA results: factor effects and interaction effects on AUC and minority-class recall involving the oversampling rate (abbr. over) and λ in terms of the base learner (abbr. learner) and training data (abbr. data). The symbol "*" indicates the interaction between two factors.

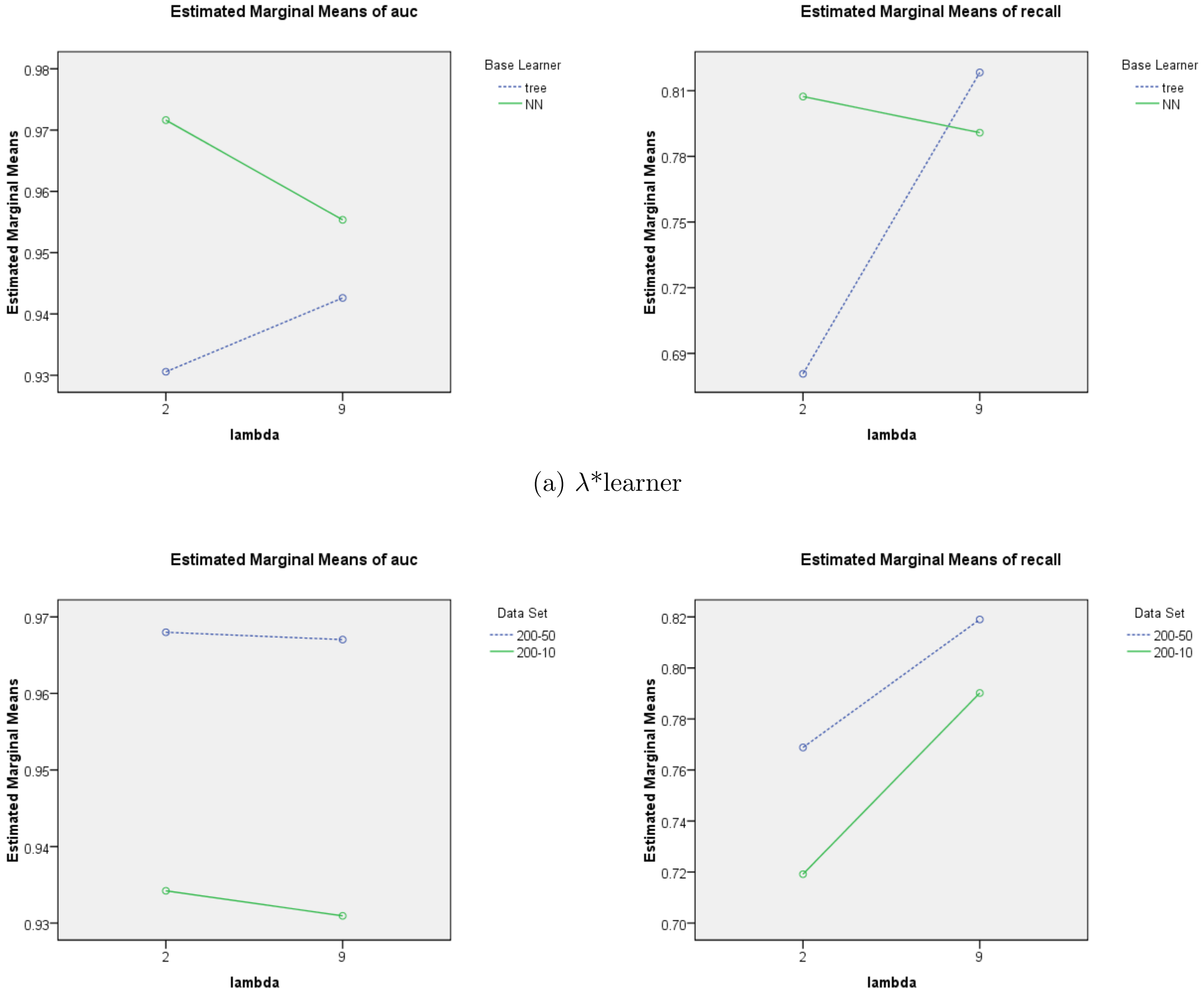
AUC	p-val	η^2	Minority-class recall	p-val	η^2
over	.955	.000	over	.087	.002
over*learner	.385	.000	over*learner	.207	.001
over*data	.166	.000	over*data	.062	.003
over*learner *data	.952	.000	over*learner *data	.034	.004
λ	.032	.006	λ	.000	.176
λ*learner	.000	.140	λ*learner	.000	.286
λ*data	.233	.000	λ*data	.010	.005
λ*learner *data	.003	.006	λ*learner *data	.025	.004

The results show that the oversampling rate does not have a significant effect on AUC and minority-class recall in general. The effects of "over" and interactions involving "over" are not significant in 7 out of 8 cases, where the p-value is larger than 0.05. Even though the interaction between "over", "learner" and "data" has a significant effect on minority-class recall, its effect size is very small ($\eta^2 = 0.004$). The weak interactions here imply that the oversampling rate always has very little effect on AdaBoost.NC regardless of what base learner and training data are used. This is a reasonable behaviour, as random oversampling is the most conservative sampling technique without losing or generating any data information. Data replication does not cause the change of decision boundaries. As long as the minority class draws comparable attention of the learning algorithm with the majority class, the oversampling rate should not affect AdaBoost.NC's performance significantly.

Different observations are obtained on λ. It presents a significant effect in terms of both AUC and minority-class recall, when varying from low to high. Especially, the interaction effect of λ*learner appears to be quite strong with a much higher η^2 value than the others (0.140 and 0.286). It means that the impact of λ depends on the base learner. However, the interaction effect of λ*data is rather weak, which is not significant on AUC and has a very small η^2 value (0.005) on minority-class recall. So, the effect of λ is not affected by the training data much. For a clear understanding of λ's effect, we further draw the plots of marginal means of performance for λ*learner and λ*data in Fig. 5.11. The plots for λ*learner*data are similar to the 2-factor ones due to its low effect size, and were omitted here for space considerations.

Fig. 5.11(a) shows different effects of λ in AdaBoost.NC between base learners. As λ varies from low to high, the tree-based AdaBoost.NC improves AUC and minority-class recall, whereas the NN-based AdaBoost.NC reduces them. The NN-based AdaBoost.NC presents better AUC than the tree-based one at both λ values. The tree-based AdaBoost.NC with a large λ improves minority-class recall greatly, which shows even better ability to recognize minority class examples than the NN-based AdaBoost.NC.

To explain why the effectiveness of AdaBoost.NC varies between base learners, a possible reason could be that a NN is less sensitive to the change of number of training data than a decision tree, which has been observed in (Japkowicz and Stephen, 2002;



(a) λ*learner

(b) λ*data

Figure 5.11: Marginal means of responses for λ*learner and λ*data.

Khoshgoftaar et al., 2010) and confirmed in our experiments. Neural networks can be thought of as a less global approach to partitioning the data space than decision trees since they get modified by each data point or a batch of data points sequentially and repeatedly through the error function (Japkowicz and Stephen, 2002). With a different training strategy, a decision tree grows based on the whole training set by counting the class frequency at each node. We conjecture that the sensitivity of a classifier to the characteristics of training examples affects the effectiveness of AdaBoost.NC in handling imbalanced data sets. This point will be further discussed in the next section by providing more empirical evidences.

From the view of training data, two response lines in each plot of Fig. 5.11(b) are nearly parallel to each other, which further confirms the weak interaction between λ and the imbalance rate of training data previously pointed out by the low η^2 value. The result is reasonable, because λ does not operate on training data directly, so its impact should not depend on the given data set greatly.

In summary, the ANOVA results suggest that AdaBoost.NC is insensitive to the oversampling rate if the minority class draws relatively equal attention of the learning algorithm to the majority class. λ is the main factor that decides the generalization of AdaBoost.NC. Its effect depends on the base learner. When the decision tree is used as the base learner, a large λ is recommended. Its effect is not affected much by the training data we tested on, because λ does not work on the data level. As a remarkable improvement over other resampling-based ensemble methods, AdaBoost.NC simply learns from the original imbalanced data without generating new minority class data or removing majority class data. It reduces the dependence of the algorithm on resampling techniques and training data, which is supported by the ANOVA experiment in this section. Remaining questions for future are the behaviour of AdaBoost.NC using other base learning algorithms and how large λ could be.

5.3 Comparisons with Other NCL Algorithms and Class Imbalance Learning Methods

So far, we have discussed the effectiveness of AdaBoost.NC under some artificial imbalanced scenarios. It is necessary to compare its performance with other state-of-art methods in real-world domains. In this section, more experiments are set up to investigate NCL algorithms, including AdaBoost.NC, CELS (Liu and Yao, 1999b) and NCCD (Chan and Kasabov, 2005b), in terms of their capabilities in dealing with class imbalance problems. Three popular Boosting-based solutions for class imbalance learning, RareBoost (Joshi et al., 2001), SMOTEBoost (Chawla et al., 2003) and JOUS-Boost (Mease et al., 2007), join the comparison.

5.3.1 Real-World Data Domains and Experimental Settings

We continue our study on fifteen real-world imbalanced data sets, which are the same as the ones discussed in chapter 3. Data information is summarized in Table 5.6. The first ten are from software engineering projects in the PROMISE data repository (Boetticher et al., 2007). The last five data sets are frequently discussed in the class imbalance learning literature.

In our experiments, we still employ C4.5 decision trees and MLP neural networks as base learners, following the settings in the previous section. We implement 7 ensemble algorithms plus a single classifier model. The seven ensemble algorithms include three NCL methods (i.e. AdaBoost.NC, CELS, NCCD), three imbalance learning methods (i.e. SMOTEBoost (Chawla et al., 2003), JOUS-Boost (Mease et al., 2007), RareBoost-1 (Joshi et al., 2001)) and the conventional AdaBoost. As before, three training strategies are applied to NCL methods, AdaBoost and the single classifier: learning from the original data, random oversampling, and random undersampling, which are denoted as “Or”, “Ov” and “Un” respectively in the following tables and analysis to simply our notations. It is worth mentioning that CELS and NCCD are not applicable to tree ensembles. λ in

Table 5.6: Summary of real-world data sets.

Data	Size	Attributes	Percentage of minority class(%)
mc2	161	39	32.29
mw1	403	37	7.69
kc3	458	39	9.38
cm1	498	21	9.83
kc2	522	21	20.49
pc1	1109	21	6.94
pc4	1458	37	12.20
pc3	1563	37	10.23
kc1	2109	21	15.45
pc2	5589	36	0.41
glass	214	9	7.94
ecoli	336	7	10.42
balance	625	4	7.84
car	1728	6	3.99
insurance	5822	85	5.98

AdaBoost.NC is set to 2 and 9, to consider both cases of encouraging diversity conservatively and aggressively. To sum up, 21 NN-based models and 15 tree-based models are constructed in total for every data set. Their settings are described as follows.

- 1-3) Single tree or NN: train a single classifier using 3 different strategies. (abbr. OrSg, OvSg, UnSg)

- 4-6) AdaBoost (Freund and Schapire, 1996): train 51 trees or NNs using AdaBoost with 3 different strategies. (abbr. OrAda, OvAda, UnAda)

- 7-12) AdaBoost.NC (Wang et al., 2010): train 51 trees or NNs using AdaBoost.NC with 3 different strategies. The penalty strength λ is set to 2 and 9. (abbr. OrNC, OvNC, UnNC followed by the λ value)

- 13-15) CELS (Liu and Yao, 1999b): train 9 NNs simultaneously using CELS (abbr. OrCELS, OvCELS, UnCELS). The scaling coefficient λ is set to 1, to allow a significant effect of diversity. Trees are not applicable. To reduce the computation cost, the ensemble size is limited to 9. Besides, it becomes more difficult to achieve smaller correlations when more networks are added in (Brown and Yao, 2001). "Winner-take-all" is used to combine the outputs as demanded by the original paper.

- 16-18) NCCD (Chan and Kasabov, 2005a): train 9 NNs using NCCD (abbr. OrNCCD, OvNCCD, UnNCCD). The scaling coefficient λ is set to 0.8. Trees are not applicable. "Winner-take-all" is used to combine the outputs as demanded by the original paper.

- 19) SMOTEBoost (Chawla et al., 2003) (abbr. SMB): it uses SMOTE (Chawla et al., 2002) to generate new minority class examples in each training epoch. The k nearest neighbour parameter is five as recommended by the original paper. Without knowing the "right" class distribution, the amount of new data in each iteration is roughly set to $|N_{maj}| - |N_{min}|$, where $|N_{maj}|$ and $|N_{min}|$ denote the numbers of majority and minority class examples respectively in the training set. This is to ensure fair comparison, considering that our method also adjusts the class ratio to one. 51 trees or NNs are built sequentially.

- 20) JOUS-Boost (Mease et al., 2007) (abbr. JSB): noise is introduced into each Boosting training epoch by applying jittering to repeated training examples. Noise with i.i.d. uniform $(-\delta\sigma_j, \delta\sigma_j)$ is added to each of the continuous attributes, where σ_j is the standard deviation of the j-th attribute, and δ is the tuning parameter to adjust the noise level. δ is fixed at 1, as suggested by the authors (Mease et al., 2007). 51 trees or NNs are built sequentially.

- 21) RareBoost-1 (Joshi et al., 2001) (abbr. RAB): it is a cost-sensitive but cost-free method. The rare class is emphasized by distinguishing TN from FN, and TP from FP at each iteration. The balance between recall and precision is thus achieved. 51 trees or NNs are built sequentially. If the conditions of $TP > FP$ and $TN > FN$ are not satisfied, the training procedure will stop early.

For an easy access, the methods and their settings are summarized in Table 5.7. A 10-fold cross-validation (CV) is performed with 10 runs.

Table 5.7: Summary of experimental methods, their acronyms and parameter settings.

No.	Acronym	Method	Parameter
1	OrSg	build a single classifier	51 trees or NNs
2	OvSg	oversampling+single classifier	51 trees or NNs
3	UnSg	undersampling+single classifier	51 trees or NNs
4	OrAda	original AdaBoost.M1 Freund and Schapire (1996)	51 trees or NNs
5	OvAda	oversampling+AdaBoost.M1	51 trees or NNs
6	UnAda	undersampling+AdaBoost.M1	51 trees or NNs
7	OrNC2	original AdaBoost.NC Wang et al. (2010)	51 trees or NNs; $\lambda = 2$
8	OrNC9	original AdaBoost.NC	51 trees or NNs; $\lambda = 9$
9	OvNC2	oversampling+AdaBoost.NC	51 trees or NNs; $\lambda = 2$
10	OvNC9	oversampling+AdaBoost.NC	51 trees or NNs; $\lambda = 9$
11	UnNC2	undersampling+AdaBoost.NC	51 trees or NNs; $\lambda = 2$
12	UnNC9	undersampling+AdaBoost.NC	51 trees or NNs; $\lambda = 9$
13	OrCELS	original CELS Liu and Yao (1999b)	9 NNs; $\lambda = 1$
14	OvCELS	oversampling+CELS	9 NNs; $\lambda = 1$
15	UnCELS	undersampling+CELS	9 NNs; $\lambda = 1$
16	OrNCCD	original NCCD Chan and Kasabov (2005a)	9 NNs; $\lambda = 0.8$
17	OvNCCD	oversampling+NCCD	9 NNs; $\lambda = 0.8$
18	UnNCCD	undersampling+NCCD	9 NNs; $\lambda = 0.8$
19	SMB	SMOTEBoost Chawla et al. (2003)	51 trees or NNs; $k = 5$; $N = \|N_{maj}\| - \|N_{min}\|$
20	JSB	JOUS-Boost Mease et al. (2007)	51 trees or NNs; $\delta = 1$
21	RAB	RareBoost-1 Joshi et al. (2001)	51 trees or NNs

N: the amount of generated data; λ: penalty strength; δ: noise level.

5.3.2 Results and Analyses

We compare AUC, minority-class recall and precision for each pair of learning models, and output their win-tie-lose numbers based on T-test at confidence level of 95% over the 15 data sets. For a clear observation, we further calculate the percentages of win-tie-lose cases on average for each model compared with the others based on the paired comparison. The full results of paired comparison can be found in the Appendix. C section. AUC describes the general ability of a learning algorithm to separate the minority and majority classes. Minority-class recall and precision provide information of which performance aspect is improved or reduced on the minority class. Separate discussions are given to tree-based and NN-based models next.

Tree-Based Models

Table 5.8 shows the average percentages of win-tie-lose cases of tree-based models in total 210 pairs of comparison (15 data sets * 14 pairs of learning methods). According to the table, we have following observations:

Table 5.8: Percentages of win(w)-tie(t)-lose(l) cases (in '%') for each experimental method in 210 pairs of comparison using tree base learners, including AUC, minority-class recall and minority-class precision. The results are based on student T-test with confidence level of 0.95. The highest percentage of win is in boldface.

Method	AUC			Minority-class recall			Minority-class precision		
	w	t	l	w	t	l	w	t	l
OrSg	12	9	79	16	31	53	39	35	26
OvSg	6	8	86	39	22	39	32	29	39
UnSg	18	12	70	74	20	6	12	14	74
OrAda	35	40	25	14	33	53	51	40	9
OvAda	33	43	24	30	29	41	43	38	19
UnAda	44	35	21	78	17	5	17	18	65
OrNC2	52	36	12	8	24	68	**60**	30	10
OrNC9	42	39	19	2	7	91	28	20	52
OvNC2	54	34	12	28	25	47	51	40	9
OvNC9	**61**	30	9	67	11	22	39	23	38
UnNC2	49	34	17	**81**	19	0	19	20	61
UnNC9	36	26	38	80	19	1	16	17	67
SMB	37	43	20	47	17	36	48	33	19
JSB	20	14	66	10	26	64	38	28	34
RAB	32	34	34	12	30	58	44	41	15

1) OvNC9 is superior to the other learning models in terms of AUC with the highest win rate of 61% and the lowest lose rate of 9%.

To understand how OvNC9 can achieve a high AUC score, let's see recall and precision for the minority class. It's not surprising that models using undersampling always produce the highest recall values, since some data information is abandoned from the majority class. However, their classification precision is sacrificed greatly, where their win rates are all lower than 20%. It means that more majority class examples are misclassified. Therefore, they do not perform well in AUC, even though more minority class examples are labelled correctly. Among the models without using undersampling, OvNC9 attains the highest win rate (67%) of minority-class recall without losing too much precision. Thus, it shows the best AUC. In other words, it balances the between-class performance very well with improved recognition rate of minority class examples.

2) Regarding other methods, OvSg presents the worst AUC with the lowest win rate, which implies that oversampling could reduce the single tree's performance. JSB does not perform well either, which is only better than the single tree models and worse than the others. SMB is better than RAB and JSB, but still worse than OvNC ones.

3) Regarding the training strategy of resampling, we can see that random oversampling does not really improve AUC of the single tree and the conventional AdaBoost. However, AdaBoost.NC integrated with oversampling improves AUC and minority-class recall greatly, especially when λ is high. It tallies with our results on the artificial data. It suggests that this combination can discriminate the minority class from the majority class better by encouraging diversity on the minority class aggressively. The overfitting problem is lessened. Random undersampling does not work well on AdaBoost.NC, because undersampling itself is able to tackle overfitting by introducing some randomness into data space. Undersampling tends to cause over-generalization with low minority-class precision obtained.

4) For the ten defect prediction data sets in our experiment, it is useful to know that, the average rate of finding defects $\approx$ 60% was reported at the 2002 IEEE Metrics panel (Shull et al., 2002) in the software engineering community. A recent work (Menzies et al., 2007) had the average recall reach 71% over some PROMISE data sets with data cleaning and feature selection applied. In our experiments, OvNC9 produces recall for the defect class higher than 60% in 5 out of 10 data sets, 2 of which exceed 71%. For the space consideration, the raw outputs were omitted here.

In summary, when the decision tree is used as the base learner, AdaBoost.NC using random oversampling is more effective than other "resampling+ensemble/single" learning methods and the state-of-art solutions for class imbalance learning. It shows better separability between the minority and majority classes and less overfits the minority class without modifying training data. A large λ is preferable.

NN-Based Models

According to Table 5.9, we have following observations for the NN-based models in total 300 pairs of comparison (15 data sets * 20 pairs of learning methods):

Table 5.9: Percentages of win(w)-tie(t)-lose(l) cases (in '%') for each experimental method in 300 pairs of comparison using NN base learners, including AUC, minority-class recall and minority-class precision. The results are based on student T-test with confidence level of 0.95. The highest percentage of win is in boldface.

Method	AUC			Minority-class recall			Minority-class precision		
	w	t	l	w	t	l	w	t	l
OrSg	43	44	24	9	18	73	45	25	30
OvSg	39	34	27	56	19	25	43	38	29
UnSg	43	36	21	70	19	11	25	26	49
OrAda	40	35	25	24	23	53	**58**	34	8
OvAda	41	38	21	39	23	38	54	33	13
UnAda	39	42	19	76	18	6	24	24	52
OrNC2	38	33	29	26	17	57	49	37	14
OrNC9	28	39	33	8	11	81	24	35	41
OvNC2	**54**	29	17	40	18	42	52	31	17
OvNC9	43	33	24	46	19	35	45	27	28
UnNC2	34	37	29	74	16	10	21	22	57
UnNC9	25	32	43	70	20	10	18	21	61
OrCELS	39	28	33	18	18	64	47	29	24
OvCELS	36	39	25	64	17	19	39	28	33
UnCELS	37	35	28	64	20	16	29	25	46
OrNCCD	26	28	46	9	15	76	16	23	61
OvNCCD	19	21	60	41	16	43	33	24	43
UnNCCD	30	31	39	**77**	11	12	11	16	73
SMB	23	30	47	20	19	61	36	37	27
JSB	2	8	90	11	16	73	38	29	33
RAB	33	36	31	22	21	57	48	38	14

1) OvNC2 presents better AUC with the highest win rate of 54% and the lowest lose rate of 17%. A large λ tends to reduce AUC, which is also observed on the artificial data

sets.

2) With respect to the single NN and NN-based AdaBoost models, they show less sensitivity to the class imbalance than the tree-based ones. For example, the NN-based OrSg wins in AUC with the rate of 43%, whereas the tree-based OrSg only has the rate of 12%. It agrees with our results on the artificial data sets and the existing findings in the literature (Japkowicz and Stephen, 2002; Khoshgoftaar et al., 2010).

3) AdaBoost.NC seems more effective in improving AUC on the real-world data sets than on the artificial ones. In section 5.2, the single NN is generally better than AdaBoost.NC, but this is not the case here. As we have explained, although the NN is less affected by the class imbalance than the decision tree, its robustness gets weaker as the training data becomes more complex (Japkowicz, 2000b). Hence, the single NN may be more suitable for simpler data sets, such as the artificial data we have shown, whereas AdaBoost.NC is a better choice for real-world problems.

4) As to the CELS and NCCD models, the CELS ones present better AUC than the NCCD ones, but are still worse than OvNC2 and OvNC9. An unexpected observation is that OvCELS achieves very good minority-class recall, which is competitive with UnCELS and better than the other methods without using undersampling. It supports our initial idea of using NCL to better recognize rare examples. CELS is more effective in improving minority-class recall than AdaBoost.NC. The possible reason is that ensemble diversity is encouraged through the error function of the neural network, which is more straightforward for training a NN than modifying the weights of training data in AdaBoost.NC.

5) For the three class imbalance learning solutions, SMB, JSB and RAB do not show any advantage in both AUC and minority-class performance. Especially, JSB appears to be the worst.

The above observations suggest that AdaBoost.NC using random oversampling can produce better overall performance than other methods, when the neural network is used as the base learner. A small λ is preferable. CELS using oversampling is very effective in finding minority class examples, which means that it less overfits the minority class without removing any data information.

5.3.3 Additional Remarks

Based on the results obtained from the real-world data, we have following remarks regarding AUC and minority-class performance for tree-based and NN-based models.

1) Combining random oversampling and NCL can improve recall of the minority class significantly without removing any data information or generating any new data. It is observed in both tree-based and NN-based models. Although random oversampling itself could cause overfitting in general, the NCL method can exploit the replicates of rare examples to tackle this issue successfully by increasing diversity of the ensemble on the classification of the overfitted class. Without applying it, the relatively small quantity of the minority class data may diminish the effect of NCL, when compared with the large size of the majority class.

2) The NCL methods seem not to work well with undersampling, because undersampling does not provide sufficient data points for NCL to generate appropriate diversity. Undersampling itself is able to make the ensemble diverse, but sacrifices the precision of predicting the minority class greatly.

3) Between two base learners, we compare OvNC9 tree ensembles and OvNC2 NN ensembles based on T-test with confidence level of 95% in Table 5.10. Both attain the best AUC values in paired comparison among the models using the same base learner. We are interested in which base learner can have the ensemble better deal with class imbalance problems. According to the table, OvNC9 produces better AUC in 10 out of 15 data sets (3 cases are significant), better minority-class recall in 13 data sets (4 cases are significant) and better minority-class precision in 7 data sets (3 cases are significant). Therefore, we conclude that the OvNC9 tree ensemble is generally a better choice than the OvNC2 NN ensemble on real-world data sets.

Another point suggested by the table is that, AdaBoost.NC enhances the learning

Table 5.10: "Mean ± standard deviation" of AUC and minority-class performance measures by OvNC9 tree ensembles and OvNC2 NN ensembles. The significantly better value is in boldface.

	AUC		**Minority-class recall**		**Minority-class precision**	
	OvNC9 trees	OvNC2 NN	OvNC9 trees	OvNC2 NN	OvNC9 trees	OvNC2 NN
mc2	0.696±0.145	0.744±0.135	0.555±0.240	0.513±0.238	0.502±0.221	0.578±0.225
mw1	0.767±0.160	0.725±0.169	0.529±0.287	0.338±0.267	0.319±0.194	0.307±0.278
kc3	0.819±0.104	0.735±0.104	**0.612±0.230**	0.312±0.204	0.331±0.143	0.271±0.178
cm1	0.783±0.089	0.730±0.111	0.517±0.219	0.369±0.236	0.254±0.099	0.220±0.131
kc2	**0.801±0.076**	0.674±0.139	0.728±0.144	0.578±0.195	**0.502±0.090**	0.389±0.150
pc1	0.874±0.057	0.841±0.064	0.515±0.165	0.520±0.193	**0.385±0.121**	0.250±0.079
pc4	0.930±0.022	0.929±0.027	**0.892±0.074**	0.663±0.122	0.495±0.057	**0.588±0.098**
pc3	0.842±0.048	0.825±0.051	**0.659±0.127**	0.456±0.124	0.334±0.067	0.368±0.086
kc1	**0.802±0.035**	0.684±0.070	0.650±0.077	0.610±0.109	**0.370±0.039**	0.279±0.063
pc2	0.803±0.169	0.749±0.152	0.098±0.180	0.076±0.169	0.075±0.162	0.035±0.102
glass	0.858±0.089	0.905±0.064	0.470±0.286	0.340±0.293	0.300±0.219	0.370±0.280
ecoli	0.929±0.043	0.945±0.033	0.820±0.172	0.651±0.221	0.518±0.115	**0.653±0.161**
balance	0.433±0.066	**0.918±0.051**	0.122±0.100	**0.504±0.182**	0.038±0.027	**0.579±0.159**
car	0.986±0.007	0.988±0.001	0.975±0.068	0.959±0.065	0.539±0.092	**0.948±0.056**
insurance	**0.704±0.007**	0.682±0.015	**0.251±0.012**	0.238±0.015	0.165±0.010	**0.184±0.018**

ability of trees more than that of NNs. It could be attributed to the way of encouraging diversity of AdaBoost.NC. It emphasizes diversity by adjusting the training subsets in essence through the penalty term that changes the weights of training examples. An indirect effect is thus made on base learners. However, NNs tend to be more tolerant to the change of training data than trees (Japkowicz and Stephen, 2002; He and Garcia, 2009; Khoshgoftaar et al., 2010), as we have already shown in the earlier analysis. Therefore, AdaBoost.NC seems less effective in dealing with class imbalance problems when using neural networks as the base learner.

The observations we have made on real-world imbalanced tasks are quite encouraging, which support our idea of using NCL to handle class imbalance problems at the beginning of this chapter. It balances the performance between classes well without losing any data information or generating new training data. Particularly, the AdaBoost.NC tree ensemble with a large λ could be a good choice.

5.3.4 Correlation Between Diversity and Generalization Performance

This subsection discusses the relationship between diversity and classification performance produced by our training methods in order to understand the role of diversity in ensemble learning better. Spearman's rank correlation coefficient is used for correlation analysis. It is a non-parametric measure of statistical dependence between two variables. Q-statistic (Yule, 1900) is used to evaluate diversity, as one of the most popular diversity measures for classification ensembles.

Table 5.11 presents the correlation coefficient ρ between Q-statistic (Q) and AUC/minority-class recall (R)/minority-class precision (P) evaluated on testing data for the 12 tree ensembles we have built in our experiments. A negative value indicates that the corresponding performance tends to decrease when ensemble diversity reduces (equivalent to an increasing Q). In other words, there is a positive effect of diversity.

Table 5.11: Rank correlation coefficients (in '%') between Q-statistic (Q) and AUC/minority-class recall (R)/minority-class precision (P) for the 12 tree ensembles on 15 real-world data sets. "avg" is the average over all data sets.

Data	Q-R	Q-P	Q-AUC
mc2	-29	-60	-57
mw1	-36	16	21
kc3	-10	-27	46
cm1	-31	-41	39
kc2	-22	27	34
pc1	-54	11	18
pc4	-5	-15	-21
pc3	-29	-41	17
kc1	-26	11	-8
pc2	-38	-43	-6
glass	-52	-33	7
ecoli	5	16	-27
balance	-41	-42	7
car	-61	-13	-4
insurance	-40	11	23
avg	-31	-15	6

According to Table 5.11, most correlation coefficient ρ's of minority-class recall present negative. It implies that a larger Q (less diversity) tends to find fewer examples from the minority class. No consistent relationship is observed in AUC and minority-class precision from the table. It could be attributed to the nature of the experiment. First, the improvement of the performance measure is small, which cannot be reflected well by the diversity measure. Second, only twelve ensembles with limited range of Q-statistic are considered in the correlation analysis. Using different learning methods may not cause a significant change of Q-statistic. In addition, the range of diversity measures could be a factor of affecting the role of diversity (Chen, 2008). Considering more ensemble methods may be useful.

5.4 Chapter Summary

This chapter presents a comprehensive experimental study of ensemble methods for class imbalance learning. The effectiveness of negative correlation learning (NCL) algorithms is explored and exploited, as a type of ensemble learning techniques that encourages diversity explicitly. The objective is to find a better solution that less depends on the settings of data-level techniques and is easier to use than algorithm-level methods. AdaBoost.NC, a newly proposed NCL algorithm in chapter 4, is studied thoroughly on both artificial and real-world imbalanced data, in order to understand the effect of diversity and how it facilitates class imbalance learning. Two other NCL algorithms (CELS and NCCD) and some state-of-art ensemble solutions for class imbalance problems are also investigated and compared to AdaBoost.NC.

Our experimental results suggest that integrating random oversampling with NCL methods can find more minority class examples without losing the overall performance when compared with other existing methods. This is achieved by producing broader and less overfitting boundaries for the minority class. Particularly, the AdaBoost.NC tree ensemble with a large λ could be a good solution with improved AUC and minority-class recall. It is not recommended to use undersampling with NCL. Apparently, our method has following merits: no data information is lost in the majority class; no data generation method is involved. Therefore, it reduces the dependence of the algorithm on resampling techniques and training data. It is empirically supported by the results in the ANOVA experiment. Besides, it works on the ensemble level, so there is no need to modify the base learning algorithm.

So far, we find a new way to deal with class imbalance problems effectively. It shows the usefulness of ensemble diversity in this type of classification tasks. This chapter only examines two-class imbalance learning problems. It is important and interesting to study multi-class tasks and see if the effectiveness of NCL would still be maintained when multiple minority and/or majority classes exist in data.

CHAPTER 6

NEGATIVE CORRELATION LEARNING FOR MULTI-CLASS IMBALANCE PROBLEMS

All the analyses and discussions so far are focused on two-class imbalance problems. However, many real-world applications that suffer from the class imbalance difficulty have more than two classes. Multi-class brings new challenges to class imbalance learning that have not caused much attention in the literature. Performance degradation in two-class imbalance learning techniques has been observed. Therefore, more investigations are necessary to explain what problems multi-class can cause and how it affects the classification performance in the presence of imbalanced data. Moreover, there is no direct and effective method specializing in multi-class imbalance problems. We would like to explore new approaches to tackling this problem.

This chapter addresses the "multi-class" issue in class imbalance learning. Section 6.2 studies the impact of multi-class on the performance of two simple resampling techniques in class imbalance scenarios. Section 6.3 extends the solving scope of AdaBoost.NC that has shown great benefits in two-class imbalance problems in chapter 5 to multi-class cases. Section 6.4 summarizes this chapter.

6.1 Introduction

Although the efforts in the class imbalance learning literature are focused on two-class problems, two-class is not the only scenario where the class imbalance problem prevails. In practice, many problem domains have more than two classes with uneven distributions, such as protein fold classification (Zhao et al., 2008; Chen et al., 2006; Tan et al., 2003) and weld flaw classification (Liao, 2008). Multi-class imbalance problems pose new challenges that are not observed in two-class problems. Zhou et al. (Zhou and Liu, 2006b) showed that dealing with multi-class tasks with different misclassification costs of classes is more difficult than with two-class ones. Further investigations are necessary to explain *what problems multi-class can cause to existing class imbalance learning techniques and how it affects the classification performance.* Such information helps us to understand the multi-class issue better, and can be utilized to develop better solutions.

Most existing imbalance learning techniques are only designed for and tested in two-class scenarios. They have been shown to be less effective or even cause a negative effect in dealing with multi-class tasks (Zhou and Liu, 2006b). Some methods are not applicable directly. Among limited solutions for multi-class imbalance problems, most attention in the literature has been devoted to class decomposition – converting a multi-class problem into a set of two-class sub-problems (Ou and Murphey, 2007). Given a c-class problem ($c > 2$), a common decomposing scheme is to choose one class labelled as positive and merge the others labelled as negative for forming a sub-problem. Each class becomes the positive class once, and thus c binary classifiers are produced for a final decision (known as one-against-all, one-vs-others) (Rifkin and Klautau, 2004). However, it aggravates imbalanced distributions (Tan et al., 2003), and combining results from classifiers that learnt from different problems can cause potential classification errors (Jin and Zhang, 2007; Valizadegan et al., 2008), as we have explained in section 2.2. It is desirable to *develop a more effective method to handle multi-class imbalance problems.*

Aiming at the above unclear issues (relating to research questions in section 1.3.2), this chapter gives a deeper insight into multi-class imbalance problems. First, we study the impact of multi-class on the performance of random oversampling and undersampling techniques commonly used in class imbalance learning, by discussing "multi-minority" and "multi-majority" cases separately. It shows that both "multi-minority" and "multi-majority" negatively affect the overall and minority-class performance. Particularly, the "multi-majority" case tends to be more harmful. Random oversampling does not help the classification and suffers from overfitting. The effect of random undersampling is weakened as there are more minority classes. It can cause a great performance reduction to majority classes when multiple majority classes exist. Neither strategy is satisfactory. Based on the results, we propose to make use of our NCL study for two-class imbalance problems in chapter 5 to handle multi-class imbalance problems. AdaBoost.NC algorithm is shown to have good performance under two-class imbalance scenarios by exploiting ensemble diversity without losing any data information. Random oversampling is integrated to guarantee sufficient number of minority class examples. As a new study of multi-class imbalance problems, the experiments in this chapter reveal that "oversampling+AdaBoost.NC" keeps the ability to better recognize minority class examples and better balances the performance across multiple classes with high G-mean without using any class decomposition schemes.

6.2 Challenges of Multi-Class Imbalance Learning

Two types of multi-class could occur to an imbalanced data set: one majority and multiple minority classes (multi-minority cases); one minority and multiple majority classes (multi-majority cases). A problem with multiple minority and multiple majority classes can be treated as the case when both types happen. For a clear understanding, we give a formal definition for each type:

- For the multi-minority case, one class has a much larger size than the average size of all classes. This class is treated as the majority class and the others are the minority classes. In other words, $|Z_{maj}| \gg \frac{|Z|}{c}$.

- For the multi-majority case, one class has a much smaller size than the average size of all classes. This class is treated as the minority class and the others are the majority classes. In other words, $|Z_{min}| \ll \frac{|Z|}{c}$.

Several interesting research questions are raised here: *Are there any differences between multiple minority and multiple majority classes? Would these two types of problem pose the same or different challenges to a learning algorithm? Which one would be more difficult to tackle? For such multi-class imbalance problems, which aspects of a problem would be affected the most by the multi-class? Would it be a minority class, a majority class or both?*

With these questions in mind, we will give separate discussions for each type under a set of artificial scenarios. For a clear understanding, two kinds of empirical analyses are conducted: 1) Spearman's rank correlation analysis, which shows the relationship between the number of classes and every evaluated performance measure, provides the evidence of the classification difficulty brought by "multi-minority" and "multi-majority". It will answer the question of *if* the difficulties exist. 2) Performance pattern analysis, which presents the performance changing tendencies of all existing classes as more classes are added into training, reveals the performance behaviours of each class by applying different training strategies. It will tell us *what kinds of* difficulties are caused to the recognition of each class and *what* the differences between the two types of multi-class are.

6.2.1 Artificial Data Sets and Experimental Settings

To have sufficient number of classes for our study, we generate some artificial imbalanced data sets by using the method in (Zhou and Liu, 2006a). In multi-minority cases, the number of minority classes is varied from 1 to 20, and only one majority class exists. Similarly, the number of majority classes is varied from 1 to 20 in multi-majority cases, and only one class is generated as the minority. Every data set is two-dimensional. Data points in each class are generated randomly from a Gaussian distribution, where the mean and standard deviation of each attribute are random real values in [0,10], and the coefficient is a random real value in [-1, +1]. In every training data set, each minority class has 10 examples, and each majority class has 100 examples. In the corresponding testing data set, each class contains 50 examples. The data generation procedure is randomly repeated 20 times for each setting with the same numbers of minority and majority classes.

In the experiments, three ensemble training methods are compared: the conventional AdaBoost that is trained from the original imbalanced data and used as a default benchmark method (abbr. OrAda); random oversampling + AdaBoost (abbr. OvAda), where all the minority classes get their examples replicated randomly until each of them has the same size as the majority class before training starts; random undersampling + AdaBoost (abbr. UnAda), where all the majority classes get rid of some examples randomly until each of them has the same size as the minority class before training starts. Every method is run 10 times on the current training data. Therefore, the result of the ensemble in the following comparisons is the average of 200 output values (20 training files*10 runs). 51 decision trees are constructed to form an ensemble. For a clear understanding of the impact of multi-class on the training methods themselves, no class decomposition scheme is applied. The tree base learner itself is able to process "multi-class" data. Applying resampling aims to understand the impact of "multi-class" on the basic strategies of dealing with imbalanced data sets and examine their robustness to "multi-class". Other class imbalance learning techniques will be considered in our future work.

As regard to the performance assessment in multi-class imbalance learning, single-class metrics can be applied directly to evaluating the performance in each class. Recall, precision, and F-measure are included in the following discussions as single-class measures. To evaluate the overall performance over all classes, the most widely used measures, AUC and G-mean, which are originally designed and used for two-class problems, have to be adapted to multi-class scenarios. Their extensions, MAUC (Hand and Till, 2001) and extended G-mean (Sun et al., 2006) are computed in our experiments accordingly.

6.2.2 Multi-Minority Cases

The correlation analysis and performance pattern analysis are conducted on the multi-minority cases in this subsection. The number of minority classes is varied from 1 to 20. The impact of multi-minority on the performance of oversampling and undersampling techniques is illustrated and analyzed in depth.

Correlation Analysis

Five performance measures and three ensemble training methods permit 15 pairwise correlations with respect to the number of minority classes. They show if multi-minority degrades the classification performance of the three ensemble training methods and which performance aspects are affected. The three single-class measures are recorded for the minority class that joins all the training sessions from 1 to 20. Table 6.1 summarizes the correlation coefficient values. The coefficient ranges in $[-1, 1]$, where a positive (negative) value indicates a monotone increasing (decreasing) relationship.

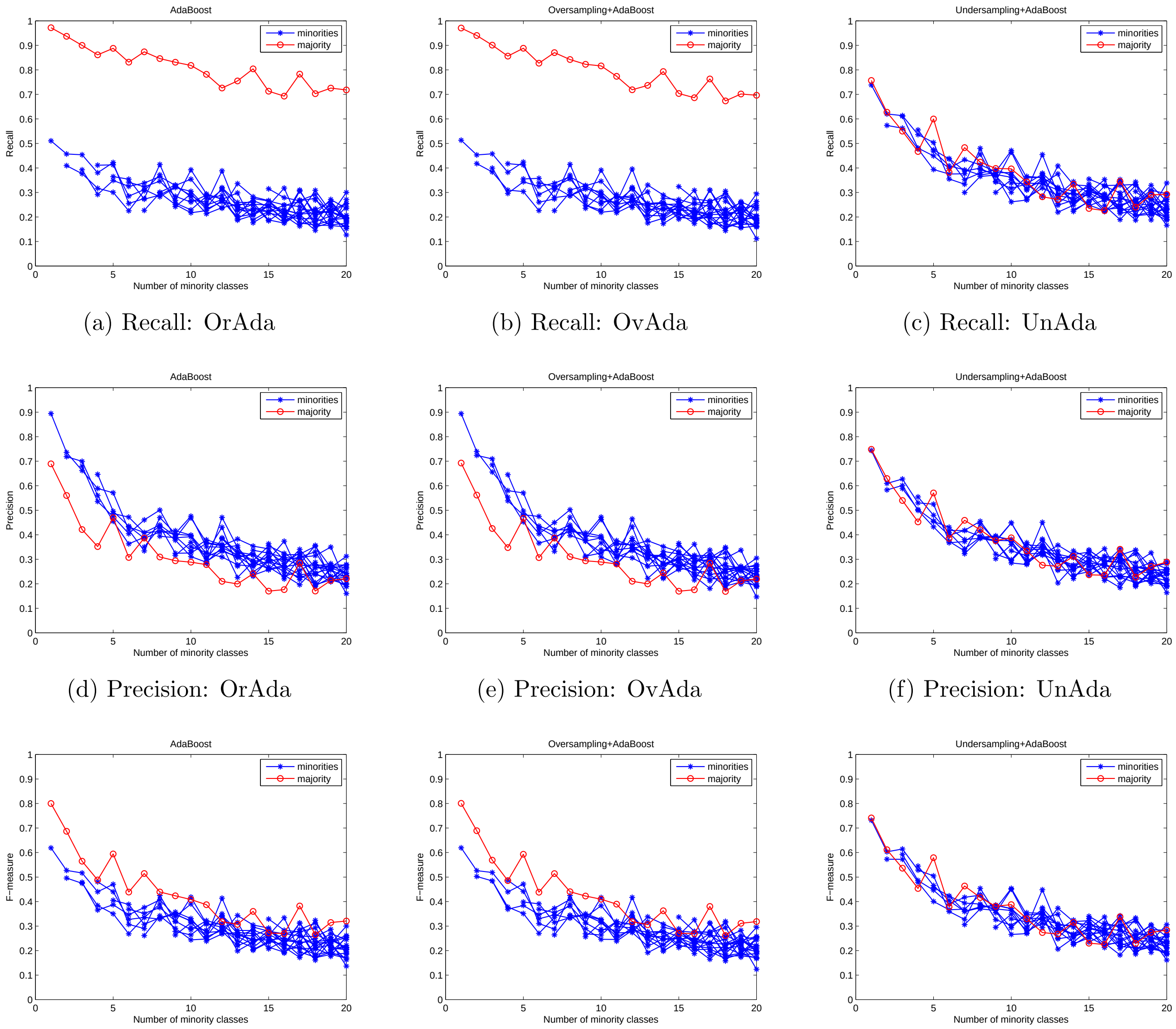
Table 6.1: Rank correlation coefficients (in %) between the number of minority classes and 5 performance measures for 3 ensemble methods. Recall, precision and F-measure are calculated for the minority class.

	Recall	Precision	F-measure	MAUC	G-mean
OrAda	-89	-94	-91	-92	-97
OvAda	-88	-93	-91	-94	-98
UnAda	-93	-93	-93	-91	-99

All pairs present very strong negative correlations. Especially, G-mean is approaching -1. It implies a strong monotone decreasing relationship between the measures and the number of minority classes. All of them are decreasing as more minority classes are added into the training data, regardless of whether resampling is applied. In other words, multi-minority reduces the performance of these ensembles consistently. Next, we will give a further investigation into the performance degradation caused by multi-minority classes.

Performance Pattern Analysis

To have a deeper insight, we illustrate the changing tendencies of single-class measures for all classes as the class number increases in Fig. 6.1. The presented pattern reveals detailed information about how the classification performance of each class is affected and the differences among ensemble methods and evaluated measures. All the following pattern plots are scaled in the same range.



(a) Recall: OrAda

(b) Recall: OvAda

(c) Recall: UnAda

(d) Precision: OrAda

(e) Precision: OvAda

(f) Precision: UnAda

(g) F-measure: OrAda

(h) F-measure: OvAda

(i) F-measure: UnAda

Figure 6.1: Single-class performance patterns among classes in multi-minority cases (x-axis: number of minority classes, y-axis: performance output).

According to Fig. 6.1, every class's performance is decreasing. No evidence shows which class suffers from more performance degradation than other classes. The classification gets equally difficult on all classes. For each class, corresponding to one curve in the plot, the measure value drops faster at the first few steps, when the minority-class number is approximately smaller than 10. As it gets larger, the reduction slows down.

Among the three performance measures, the drop of precision (Fig. 6.1(d)(e)) is more severe than that of recall (Fig. 6.1(a)(b)) in OrAda and OvAda. Precision is the main cause of the decrease in F-measure. The reason is that multi-minority increases the risk of predicting an example into a wrong class. As to recall, it seems that the difficulty of recognizing examples within each class is less affected by multi-minority as compared to precision, because the proportion of each class of data in the whole data set is hardly changed by adding a small class. For OvAda, although oversampling increases the quantity of minority class examples to make every class have the same size, the class distribution in data space is still imbalanced, which is dominated by the majority class. In UnAda, each class is reduced to have a small size. Adding minority classes changes the proportion of each class significantly. It explains the observation that UnAda's recall (Fig. 6.1(c)) presents higher sensitivity to multi-minority than the recall produced by OrAda and OvAda (Fig. 6.1(a)(b)).

Among the three ensemble methods, OrAda and OvAda have similar performance patterns, where the majority class obtains higher recall and F-measure than the minority classes, but lower precision values. Oversampling does not alleviate the multi-class problem. In UnAda, undersampling counteracts the performance differences among classes. During the first few steps, UnAda presents better recall and F-measure on minority classes (Fig. 6.1(c)(i)) than those of OrAda and OvAda (Fig. 6.1(a)(b)(g)(h)). From this point of view, it seems that using undersampling might be a better choice. However, its advantage is weakened as more minority classes join the training. When the class number reaches 20, three ensemble algorithms have very similar minority-class performance. The reason could be that undersampling explicitly empties some space for classifying minority classes by removing examples from the majority class region. When there is only one minority class, a classifier is very likely to assign the space to this class. When there are many minority classes, they have to share the same space. Hence, the effect of undersampling is reduced. Undersampling seems to be more sensitive to multi-minority. For this consideration, it would be better to expand the classification area for each minority class, instead of shrinking the majority class. To achieve this goal, advanced techniques should be applied to improving classification generalization over minority classes.

6.2.3 Multi-Majority Cases

We proceed with the same analyses for the multi-majority cases, where the number of majority classes is varied from 1 to 20. The impact of multi-majority is studied.

Correlation Analysis

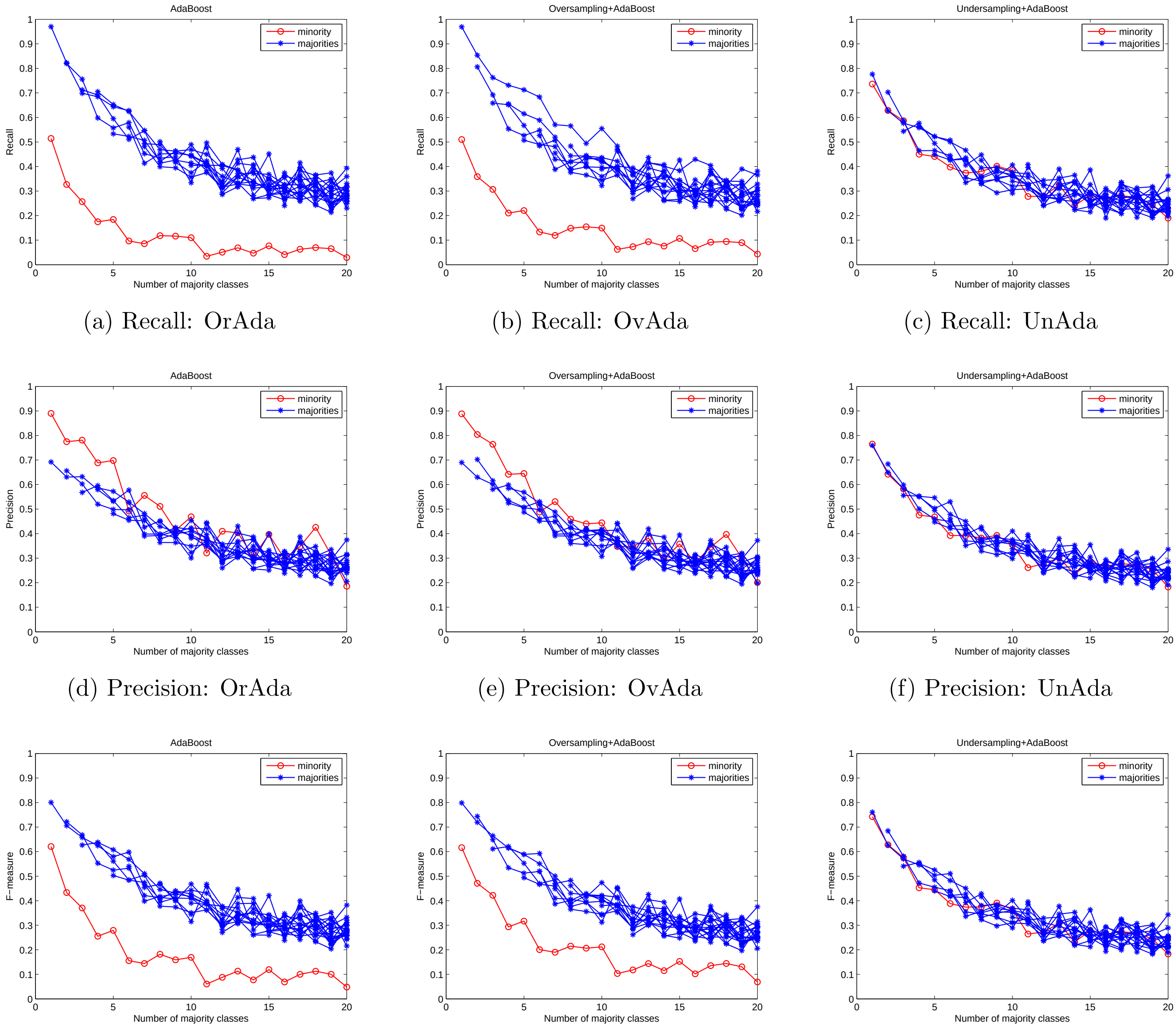
Table 6.2 summarizes the correlation coefficients in multi-majority cases. Single-class performance measures are recorded for the only minority class of each data set. Similar to the multi-minority cases, all coefficient values between five performance measures and the number of majority classes suggest strong negative correlations, which indicate a monotone decreasing relationship. All three ensemble training methods suffer from performance reduction caused by “multi-majority”.

Table 6.2: Rank correlation coefficients (in %) between the number of majority classes and 5 performance measures for 3 ensemble methods. Recall, precision and F-measure are calculated for the minority class.

	Recall	Precision	F-measure	MAUC	G-mean
OrAda	-79	-89	-85	-92	-97
OvAda	-84	-93	-86	-92	-97
UnAda	-92	-94	-94	-95	-99

Performance Pattern Analysis

To gain more information, the changing tendencies of single-class measures for each class along with the increase of the number of majority classes are shown in Fig. 6.2 within the same axis scale.



(a) Recall: OrAda (b) Recall: OvAda (c) Recall: UnAda

(d) Precision: OrAda (e) Precision: OvAda (f) Precision: UnAda

(g) F-measure: OrAda (h) F-measure: OvAda (i) F-measure: UnAda

Figure 6.2: Single-class performance patterns among classes in multi-majority cases (x-axis: number of majority classes, y-axis: performance output).

Among the classes in each plot, adding majority classes makes the recognition of examples of each class (i.e. recall presented in Fig. 6.2(a)-(c)) equally difficult. In

OrAda and OvAda, minority-class precision drops faster than that of the majority classes (Fig. 6.2(d)(e)), because the large quantity of new majority class examples overwhelm the minority class even more. Minority class examples are more likely to be misclassified than before compared to majority class examples.

All performance measures present a drastic decrease. Especially in recall plots of OrAda and OvAda (Fig. 6.2(a)(b)), more and more majority class examples take the recognition rate of the minority class down to nearly 0. For every existing majority class, adding more majority classes can make it appear to be in minority. Therefore, the recall of majority classes also shows a fast drop.

Among the three ensemble methods, UnAda produces better minority-class F-measure than OrAda and OvAda, but the recall of majority classes is sacrificed greatly. It causes the concern that using undersampling will lose too much data information when multiple majority classes exist, and can lead to severe performance reduction in majority classes.

In summary, 1) between multi-minority and multi-majority, the multi-majority case seems to cause more difficulties to OrAda and OvAda than the multi-minority case. They present much worse minority-class performance in Fig. 6.2(g)(h) compared to Fig. 6.1(g)(h). This is because adding majority class examples aggravates the imbalanced situation. 2) In OrAda and OvAda, no new information is introduced into the minority class to facilitate the classification. Overfitting minority-class regions happens with low recall and high precision values when compared with those measures obtained from the majority classes. Oversampling does not help. 3) UnAda performs the same under multi-minority and multi-majority cases due to undersampling. In the multi-minority case, it can be sensitive to the class number; in the multi-majority case, there is a high risk of sacrificing too much majority-class performance.

6.3 AdaBoost.NC for Multi-Class Imbalance Problems

Armed with a better understanding of multi-class imbalance problems, this section introduces a simple and effective ensemble learning method without using class decomposition. To make it effective, the fundamental issue that needs to be resolved is the discrimination difficulty between and within minority and majority classes. The presence of multiple minority classes increases the data complexity. In addition to more complex data distributions, the presence of multiple majority classes makes a data set even more imbalanced. Balancing the performance among classes appropriately is important. Suggested by the analysis in section 6.2, we aim at a method that can improve the generalization performance of oversampling by focusing on minority classes, instead of shrinking majority classes through undersampling, so that less data information will be lost and it will be less sensitive to multi-minority.

In chapter 5, we found that the "random oversampling + AdaBoost.NC" tree ensemble is effective in handling two-class imbalance problems. It shows a good recognition rate for the minority class and balances the performance between minority and majority classes well. Besides, its training strategy is flexible and simple without removing any training data. For the above reasons, we extend the study to multi-class cases in this section. The main research question here is *whether AdaBoost.NC is still effective in solving multi-class imbalance problems*. In order to answer the question and find out if class decomposition is necessary, we compare it with several ensemble methods in cases of using and not using class decomposition.

6.3.1 Data Sets and Experimental Settings

In the experiments, we evaluate AdaBoost.NC, the conventional AdaBoost and SMOTEBoost methods on 12 classification benchmark problems from the UCI repository (Frank and Asuncion, 2010). Each data set has more than two classes. At least one of them is significantly smaller than one of the others. The data information with class distributions is summarized in Table 6.3.

Table 6.3: Summary of benchmark data sets.

Data	Class	Size	Distribution
New-thyroid	3	215	150/35/30
Balance	3	625	49/288/288
Car	4	1728	1210/384/69/65
Nursery	4	12958	4320/328/4266/4044
Glass	6	214	70/76/17/13/9/29
Annealing	5	898	8/99/684/67/40
Solarflare2	6	1066	147/211/239/95/43/331
Page	5	5473	4913/329/28/88/115
Ecoli	5	327	143/77/52/35/20
Cleveland	5	303	164/55/36/35/13
Yeast	10	1484	463/429/244/163/51/44/35/30/20/5
Satimage	6	6435	1533/703/1358/626/707/1508

We construct 12 ensemble models for comparison for each data set. Six of them are trained directly from the multi-class data, including AdaBoost (OrAda) used as the baseline model, "random oversampling + AdaBoost" (OvAda)", "random undersampling + AdaBoost" (UnAda), "random oversampling + AdaBoost.NC" with $\lambda = 2$ (OvNC2), "random oversampling + AdaBoost.NC" with $\lambda = 9$ (OvNC9) and SMOTEBoost (Chawla et al., 2003) (SMB). The other six models use exactly the same ensemble algorithms, but work with the "one-against-all" (OAA) scheme of class decomposition – the most frequently used scheme in the multi-class imbalance learning literature. In this group of models, each trains a set of binary classifiers for every class, which will then be combined for a final decision. We adopt the combining strategy used in (Liao, 2008), which outputs the class whose corresponding binary classifier produces the highest value of belongingness among all. These models are denoted by "-d", to indicate that OAA is used.

With respect to parameter settings, λ in AdaBoost.NC controls the strength of encouraging diversity. Our earlier study in chapter 4 showed that $\lambda = 2$ is a relatively conservative setting to show if AdaBoost.NC can make a performance improvement, and

$\lambda = 9$ boosts diversity aggressively (Wang and Yao, 2010). For a full understanding, both values are considered. Applying random oversampling is necessary for AdaBoost.NC not to ignore the minority class based on our results in chapter 5. For SMOTEBoost, the nearest neighbour parameter k is set to 5, the most accepted value in the literature. The amount of new data for a class c is roughly the size difference between the largest class and class c, considering that other models also adjust the ratio between classes to one. C4.5 decision tree is employed as the base learner. Each ensemble consists of 51 trees. We perform a 5-fold cross-validation (CV) with 10 runs instead of the traditional 10-fold CV, to avoid the situation where a fold of data does not contain any examples from the minority class.

Regarding the model assessment, MAUC (Hand and Till, 2001) and extended G-mean (Sun et al., 2006) are used to evaluate the overall performance of the 12 ensemble models as in the previous section. In terms of each class, recall and precision are recorded as the single-class performance measures, for a better understanding of how an overall performance improvement or degradation happens. Student T-test with 95% confidence level is conducted for the significance analysis.

6.3.2 Ensemble Algorithms for Multi-Class Imbalance Problems

In this section, we study the performance of ensemble algorithms without using OAA and the ones using OAA with respective discussions. Based on the observations and analysis, an improved combination strategy for OAA-based ensembles is then proposed. Finally, we show whether class decomposition is necessary for multi-class imbalance learning.

Ensemble Models without Using OAA

Tables 6.4 - 6.6 summarize the paired comparison results of the ensemble models without using OAA. Each entry in the tables is in a win-tie-lose form over 12 data sets for a pair of models. Table 6.4 compares the overall performance measures MAUC and G-mean. According to the MAUC results (upper triangle), AdaBoost.NC does not show much advantage over other methods. Concretely, OrAda performs the best with more "win" cases in each pair of comparisons. UnAda performs the worst with the most "lose" cases. OvNC2 and OvAda are comparable to each other, which are better than OvNC9 but worse than SMB. However, different observations are obtained in G-mean (lower triangle): OrAda is the worst with the most "lose" cases; OvNC9 achieves the most wins.

Table 6.4: Ensembles without using OAA: paired comparison of MAUC (upper triangle) and G-mean (lower triangle) based on T-test with confidence level of 0.95 over 12 data sets. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrAda	OvAda	UnAda	OvNC2	OvNC9	SMB
OrAda	-	5-7-0	11-0-1	4-8-0	10-2-0	6-4-2
OvAda	2-9-1	-	9-1-2	2-9-1	9-3-0	2-6-4
UnAda	6-3-3	4-5-3	-	2-0-10	3-0-9	0-2-10
OvNC2	3-8-1	1-11-0	4-4-4	-	8-2-2	1-5-6
OvNC9	6-4-2	5-5-2	4-5-3	5-5-2	-	0-5-7
SMB	5-7-0	4-8-0	4-6-2	4-8-0	2-7-3	-

It is interesting to observe that the conventional AdaBoost without using any specific rebalancing technique is good at MAUC and bad at G-mean. It is known that AdaBoost itself cannot handle class imbalance problems very well. It is sensitive to imbalanced distributions (Joshi et al., 2002; Sun et al., 2007). Meanwhile, our experiments in the previous section show that it suffers from multi-class difficulties significantly. It means that low G-mean of AdaBoost results from its low recognition rates on the minority classes and high MAUC is probably attributed to the relatively good discriminability between the majority classes. The other ensemble methods, AdaBoost.NC, SMOTEBoost and resampling-based AdaBoost, seem to improve G-mean more than MAUC.

To explain this observation, let's review the definitions of MAUC and G-mean. G-mean is the geometric mean of recall over all classes. If any single class receives very low recall, it will take the G-mean value down. It can tell us how well a classifier can balance the recognition among different classes. A high G-mean guarantees that no class is ignored. MAUC assesses the average ability of separating any pair of classes. A high MAUC implies a classifier is good at separating most pairs, but it is still possible that some classes are hard to be distinguished from each other. G-mean is more sensitive to single-class performance than MAUC. From this point of view, it may suggest that those ensemble solutions for class imbalance learning, especially OvNC9, can better recognize examples from the minority classes, but are not good at discriminating between some majority classes. To confirm our explanations, we look into single-class performance next.

Considering the existence of multiple minority and majority classes, we only discuss recall and precision for the smallest class and the largest class, which should be the most typical ones in the data set. The win-tie-lose results are presented in Tables 6.5 and 6.6.

Table 6.5: Ensembles without using OAA: paired comparison of recall (upper triangle) and precision (lower triangle) of the smallest class based on T-test with confidence level of 0.95 over 12 data sets. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrAda	OvAda	UnAda	OvNC2	OvNC9	SMB
OrAda	-	0-10-2	0-4-8	0-9-3	0-3-9	0-7-5
OvAda	0-10-2	-	0-2-10	0-11-1	0-3-9	0-10-2
UnAda	2-2-8	3-1-8	-	9-3-0	5-7-0	7-5-0
OvNC2	0-9-3	1-10-1	8-2-2	-	0-5-7	1-10-1
OvNC9	2-5-5	4-1-7	6-4-2	2-5-5	-	7-5-0
SMB	0-9-3	3-9-0	8-3-1	2-10-0	6-4-2	-

Table 6.5 shows the comparison for the smallest class. Not surprisingly, UnAda performs the best in recall but produces the worst precision because of the loss of a large amount of data information. OvNC9 comes the second in recall with more wins and produces better precision than UnAda. SMB's recall is competitive with that of OvNC2, but worse than OvNC9's. SMB has better precision than OvNC9. OvNC9 produces better recall than OvNC2, which implies that a large λ can further generalize the performance of AdaBoost.NC on the minority class, so that more minority class examples are identified.

Table 6.6 shows the comparison for the largest class. Because of the performance

Table 6.6: Ensembles without using OAA: paired comparison of recall (upper triangle) and precision (lower triangle) of the largest class based on T-test with confidence level of 0.95 over 12 data sets. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrAda	OvAda	UnAda	OvNC2	OvNC9	SMB
OrAda	-	4-7-1	10-2-0	4-7-1	9-3-0	5-7-0
OvAda	1-8-3	-	10-2-0	0-11-1	10-2-0	4-8-0
UnAda	3-4-5	4-3-5	-	0-2-10	0-3-9	0-2-10
OvNC2	1-10-1	1-11-0	5-4-3	-	10-2-0	4-7-1
OvNC9	3-8-1	4-7-1	5-5-2	4-7-1	-	0-4-8
SMB	4-7-1	4-8-0	5-6-1	5-7-0	2-6-4	-

trade-off between minority and majority classes, OrAda performs the best in recall and UnAda performs the worst, which is opposite to the observation on the smallest class. OvNC9 and SMB are not good at recall of this class, which are only better than UnAda and worse than the others. Due to the trade-off between recall and precision, OvNC9 and SMB presents comparatively better precision values than the others. Between OvNC9 and SMB, the former presents worse recall than the latter, but slightly better precision with more wins.

The observations on the smallest and largest classes explains that the good G-mean of OvNC9 results from the greater improvement in recall of the minority classes than the recall reduction of the majority classes. Its ineffectiveness in MAUC should be caused by the relatively poor performance in the majority classes. Based on the above results, we conclude that AdaBoost.NC with a large λ is helpful for recognizing more minority class examples with high recall and capable of balancing the performance across different classes with high G-mean. From the view of MAUC and majority-class performance, it could lose some learning ability to separate majority classes. In addition, SMOTEBoost presents quite stable overall performance in both MAUC and G-mean, and is not bad at minority-class performance.

Ensemble Models Using OAA

The experimental results of the ensembles using OAA are summarized in Tables 6.7 - 6.9. Table 6.7 compares the overall performance. We observe that except SMB-d presenting better MAUC than the others with more wins, no single class imbalance learning method actually outperforms the conventional AdaBoost (i.e. OrAda-d) consistently in both MAUC and G-mean. UnAda-d shows the worst MAUC and G-mean. OvAda-d, OvNC2-d and OvNC9-d are comparable to OrAda-d in terms of MAUC. OrAda-d, OvAda-d, OvNC2-d and SMB-d show competitive G-mean with each other and better G-mean than OvNC9-d. These results are different from what we have observed in the cases without using OAA, where OvNC9 yields the best G-mean. It seems that class imbalance techniques are not very effective when working with the OAA scheme. SMOTEBoost appears to be relatively stable with good MAUC and G-mean compared to AdaBoost.NC and resampling techniques.

Table 6.7: Ensembles using OAA: paired comparison of MAUC (upper triangle) and G-mean (lower triangle) based on T-test with confidence level of 0.95 over 12 data sets. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrAda-d	OvAda-d	UnAda-d	OvNC2-d	OvNC9-d	SMB-d
OrAda-d	-	6-1-5	11-0-1	5-1-6	6-1-5	3-2-7
OvAda-d	0-12-0	-	10-0-2	2-3-7	5-3-4	2-3-7
UnAda-d	3-3-6	2-3-7	-	1-1-10	4-0-8	2-0-10
OvNC2-d	1-11-0	0-11-1	7-2-3	-	6-1-5	3-2-7
OvNC9-d	2-5-5	2-4-6	3-5-4	2-4-6	-	3-3-6
SMB-d	1-11-0	1-10-1	6-3-3	2-9-1	6-4-2	-

As to the single-class performance in the smallest class presented in Table 6.8, UnAda-d performs the best in recall but the worst in precision. The other five models are comparable to each other. They do not show much advantage in identifying minority class examples. Similar observations are obtained for the largest class according to Table 6.9. It is not clear which method is significantly better than any of the others on the majority-class performance. The above results tell us that AdaBoost.NC, SMOTEBoost and resampling techniques exhibit ineffectiveness in both minority and majority classes when compared with the conventional AdaBoost. Neither type of classes is better recognized. When the OAA scheme is applied to handling multi-class, they do not bring a consistent improvement.

Table 6.8: Ensembles using OAA: paired comparison of recall (upper triangle) and precision (lower triangle) of the smallest class based on T-test with confidence level of 0.95 over 12 data sets. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrAda-d	OvAda-d	UnAda-d	OvNC2-d	OvNC9-d	SMB-d
OrAda-d	-	0-11-1	3-2-7	0-11-1	4-5-3	0-12-0
OvAda-d	1-11-0	-	3-2-7	1-9-2	4-5-3	1-10-1
UnAda-d	1-3-8	2-1-9	-	7-1-4	7-3-2	7-1-4
OvNC2-d	1-10-1	0-11-1	8-2-2	-	4-6-2	0-12-0
OvNC9-d	2-5-5	1-6-5	7-5-0	1-7-4	-	2-5-5
SMB-d	1-10-1	0-12-0	8-2-2	2-10-0	5-6-1	-

Table 6.9: Ensembles using OAA: paired comparison of recall (upper triangle) and precision (lower triangle) of the largest class based on T-test with confidence level of 0.95 over 12 data sets. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrAda-d	OvAda-d	UnAda-d	OvNC2-d	OvNC9-d	SMB-d
OrAda-d	-	2-9-1	8-4-0	4-7-1	9-2-1	4-8-0
OvAda-d	1-9-2	-	8-4-0	1-10-1	9-2-1	4-8-0
UnAda-d	4-5-3	4-5-3	-	1-4-7	4-4-4	2-4-6
OvNC2-d	1-11-0	2-10-0	3-6-3	-	9-2-1	4-8-0
OvNC9-d	4-6-2	4-6-2	2-6-4	4-5-3	-	1-3-8
SMB-d	1-10-1	1-10-1	3-5-4	0-11-0	2-6-4	-

According to our results here, AdaBoost.NC does not show any significant improvement in minority-class and overall performance when working with the class decomposition scheme in multi-class imbalance scenarios, although it showed good classification ability in dealing with two-class imbalance problems (Wang and Yao, 2011). A possible reason for its poor performance could be that the combining step of OAA messes up the individual results. Without using OAA, AdaBoost.NC receives and learns from complete data information of all classes, which allows the algorithm to consider the difference among classes during learning with full knowledge. The OAA scheme, however, decomposes the whole problem, which makes AdaBoost.NC learn from multiple binary sub-problems with partial data knowledge. The relative importance between classes could not be taken into consideration. Even if AdaBoost.NC can be good at handling each sub-problem, their combination does not guarantee good performance for the whole problem. Therefore, it may be not a wise idea to integrate class decomposition with class imbalance techniques without considering the class distribution globally. A better combining method for class decomposition schemes is needed.

Ensemble Models Using OAA with an Improved Combination Method

To take into account the class information of the whole data set, we improve the combination method of OAA in this section by using a weighted combining rule. Instead of the traditional way of treating the outputs of binary classifiers equally (Liao, 2008), we assign them different weights, decided by the size of each class. For any example x, its belongingness value of class i from the i-th binary classifier is multiplied by the inverse of its imbalance rate. The imbalance rate is defined as the proportion of this class of data within the whole data set. The final decision of OAA will be the class receiving the highest belongingness value among all after adjusted by the weights.

We apply the same ensemble methods in the previous sections on the 12 UCI data sets. Six ensemble models are constructed. They are denoted by "-dw", indicating that class decomposition with a weighted combination is used. The win-tie-lose comparison among the six models is summarized in Tables 6.10 - 6.12.

It is encouraging to observe that the ineffectiveness of AdaBoost.NC used with OAA is rectified by the weighted combination method in terms of G-mean and minority-class recall. The results are similar to the ones without using OAA: 1) OrAda and SMB perform the best in MAUC, and UnAda performs the worst. 2) OvNC9 produces the best G-mean with the most wins. UnAda gives the worst G-mean which is a different observation from

Table 6.10: Ensembles using OAA with the weighted combination: paired comparison of MAUC (upper triangle) and G-mean (lower triangle) based on T-test with confidence level of 0.95 over 12 data sets. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrAda-dw	OvAda-dw	UnAda-dw	OvNC2-dw	OvNC9-dw	SMB-dw
OrAda-dw	-	6-4-2	12-0-0	6-2-4	6-1-5	3-4-5
OvAda-dw	0-11-1	-	12-0-0	5-1-6	7-0-5	5-0-7
UnAda-dw	2-2-8	2-1-9	-	0-1-11	2-0-10	0-1-11
OvNC2-dw	0-12-0	0-12-0	9-1-2	-	7-0-5	4-0-8
OvNC9-dw	6-3-3	5-4-3	9-1-2	4-5-3	-	4-1-7
SMB-dw	1-11-0	0-12-0	9-1-2	0-12-0	3-4-5	-

Table 6.11: Ensembles using OAA with the weighted combination: paired comparison of recall (upper triangle) and precision (lower triangle) of the smallest class based on T-test with confidence level of 0.95 over 12 data sets. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrAda-dw	OvAda-dw	UnAda-dw	OvNC2-dw	OvNC9-dw	SMB-dw
OrAda-dw	-	0-11-1	1-0-11	0-10-2	0-4-8	0-10-2
OvAda-dw	0-12-0	-	1-0-11	0-11-1	0-6-6	0-10-2
UnAda-dw	2-0-10	2-0-10	-	11-0-1	8-3-1	10-1-1
OvNC2-dw	0-10-2	0-10-2	10-0-2	-	0-7-5	1-9-2
OvNC9-dw	3-2-7	3-1-8	9-2-1	3-4-5	-	7-5-0
SMB-dw	0-10-2	0-9-3	9-1-2	2-8-2	6-3-3	-

Table 6.12: Ensembles using OAA with the weighted combination: paired comparison of recall (upper triangle) and precision (lower triangle) of the largest class based on T-test with confidence level of 0.95 over 12 data sets. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrAda-dw	OvAda-dw	UnAda-dw	OvNC2-dw	OvNC9-dw	SMB-dw
OrAda-dw	-	2-8-2	11-1-0	4-7-1	10-2-0	9-3-0
OvAda-dw	1-9-2	-	11-1-0	2-9-1	10-2-0	7-5-0
UnAda-dw	5-4-3	4-5-3	-	0-1-11	0-3-9	0-2-10
OvNC2-dw	1-11-0	2-10-0	3-4-5	-	10-2-0	4-7-1
OvNC9-dw	4-8-0	5-7-0	3-6-3	4-8-0	-	0-4-8
SMB-dw	2-8-2	2-9-1	3-4-5	3-7-2	0-6-6	-

the cases without using OAA. 3) Except UnAda, OvNC9 produces better minority-class recall than the other models. It has better minority-class precision than UnAda. 4) OvNC9 loses some performance on the majority class with a lower recall than the others except UnAda.

In summary, AdaBoost.NC with a large λ can find more minority class examples with a higher recall and better balance the performance across different classes with a higher G-mean than other methods. Its performance on the majority class is sacrificed to some extent, leading to unsatisfactory MAUC. SMOTEBoost is a comparatively stable algorithm with good overall and minority-class performance.

Is Class Decomposition Necessary?

The discussions in this subsection aim to answer the question of whether it is necessary to use class decomposition for handling multi-class imbalance problems. We compare the overall and minority-class performance produced by AdaBoost.NC with $\lambda = 9$ and SMOTEBoost methods without using OAA (i.e. OvNC9 and SMB) with the performance produced by those using OAA with the weighted combination method (i.e. OvNC9-dw and SMB-dw). We choose OvNC9 and SMB, because AdaBoost.NC is better at G-mean and minority-class recall and SMOTEBoost is better at MAUC. Raw performance outputs from 12 data sets are shown in Table 6.13. Values in boldface indicate "significantly better" between OvNC9 (SMB) and OvNC9-dw (SMB-dw).

According to the table, no consistent difference is observed between OvNC9 and OvNC9-dw in all three performance measures. In most cases, they present competitive measure values with each other. OvNC9-dw shows slightly better G-mean with more wins. The same happens between SMB and SMB-dw. It suggests that whether to apply OAA, the most commonly used technique in the literature, does not affect class imbalance learning methods much. Learning from the whole data set directly is sufficient for them to achieve good MAUC and G-mean and find minority-class examples effectively. Therefore, we conclude that using class decomposition is not necessary to tackle multi-class

Table 6.13: "Mean ± standard deviation" of MAUC, G-mean and minority-class recall by AdaBoost.NC with $\lambda = 9$ and SMOTEBoost without using OAA (i.e. OvNC9 and SMB) and using OAA with the weighted combination (i.e. OvNC9-dw and SMB-dw). Recall is computed for the smallest class of each data set. Values in boldface indicate "significantly better" between OvNC9 (SMB) and OvNC9-dw (SMB-dw).

MAUC				
	OvNC9	OvNC9-dw	SMB	SMB-dw
New-thyroid	**0.983±0.013**	0.973±0.003	0.988±0.014	0.988±0.003
Balance	0.704±0.037	0.703±0.003	**0.703±0.027**	0.633±0.004
Car	**0.982±0.005**	0.980±0.001	0.994±0.003	**0.997±0.000**
Nursery	0.995±0.001	**0.998±0.000**	0.999±0.000	0.999±0.000
Glass	0.876±0.037	0.881±0.009	0.925±0.030	0.924±0.009
Annealing	**0.986±0.009**	0.975±0.003	0.984±0.018	0.981±0.004
Solarflare2	0.866±0.020	**0.901±0.003**	0.890±0.015	0.891±0.002
Page	**0.989±0.004**	0.984±0.001	**0.989±0.005**	0.973±0.002
Ecoli	0.952±0.020	0.957±0.002	0.954±0.019	**0.963±0.004**
Cleveland	0.727±0.046	**0.766±0.004**	0.764±0.040	0.767±0.007
Yeast	0.810±0.020	**0.857±0.004**	0.831±0.021	**0.847±0.003**
Satimage	0.984±0.002	**0.990±0.000**	0.991±0.001	0.992±0.000
G-mean				
	OvNC9	OvNC9-dw	SMB	SMB-dw
New-thyroid	0.927±0.052	0.915±0.056	0.934±0.060	0.940±0.057
Balance	0.321±0.173	0.319±0.180	0.000±0.000	0.000±0.000
Car	**0.924±0.024**	0.897±0.038	0.928±0.033	**0.944±0.031**
Nursery	0.954±0.006	**0.967±0.006**	0.992±0.004	**0.996±0.003**
Glass	0.571±0.278	0.578±0.249	0.561±0.343	0.508±0.344
Annealing	0.823±0.310	0.895±0.191	0.764±0.341	0.854±0.258
Solarflare2	0.486±0.112	**0.540±0.096**	0.520±0.120	0.514±0.094
Page	0.912±0.031	0.920±0.026	0.860±0.048	0.871±0.052
Ecoli	0.776±0.069	0.790±0.062	0.798±0.052	0.803±0.059
Cleveland	0.117±0.160	0.075±0.144	0.009±0.066	0.000±0.000
Yeast	0.237±0.270	0.190±0.257	0.140±0.240	0.060±0.164
Satimage	0.872±0.011	**0.881±0.009**	0.895±0.011	0.898±0.010
Minority-Class Recall				
	OvNC9	OvNC9-dw	SMB	SMB-dw
New-thyroid	0.897±0.134	0.910±0.117	0.900±0.121	0.906±0.135
Balance	0.144±0.112	0.150±0.112	0.000±0.000	0.000±0.000
Car	0.980±0.043	**0.997±0.021**	0.967±0.058	0.973±0.057
Nursery	0.985±0.020	0.992±0.018	0.983±0.017	**0.993±0.012**
Glass	**0.990±0.070**	0.930±0.202	0.910±0.218	0.860±0.248
Annealing	0.790±0.351	0.870±0.263	0.730±0.380	0.850±0.307
Solarflare2	0.400±0.193	0.456±0.190	0.275±0.142	**0.353±0.169**
Page	0.974±0.076	0.985±0.052	0.881±0.161	0.822±0.187
Ecoli	0.820±0.182	0.860±0.176	0.870±0.169	0.880±0.153
Cleveland	0.167±0.212	0.197±0.246	0.030±0.119	0.030±0.104
Yeast	0.097±0.117	0.070±0.101	**0.056±0.092**	0.023±0.058
Satimage	0.721±0.042	**0.800±0.037**	0.692±0.050	0.709±0.040

imbalance problems.

6.4 Chapter Summary

This chapter aims to address the multi-class issue in class imbalance learning and obtain a more effective and direct solution. First, we have studied the new challenges posed by multi-class to understand what problems it can cause. Two types of multi-class imbalance problems, i.e. the multi-minority and multi-majority cases, are analyzed in depth. For each type, we examine overall and minority-class performance of three ensemble methods based on the correlation analysis and performance pattern analysis. Both types show strong negative correlations with five performance measures, i.e. MAUC, G-mean, minority-class recall, minority-class precision and minority-class F-measure. It implies that the performance decreases as the number of imbalanced classes increases. The results from the performance pattern analysis show that the multi-majority case tends to cause more performance degradation than the multi-minority case, because the imbalance rate gets more severe. Oversampling does not help the classification, and causes overfitting to the minority classes with low recall and high precision values. Undersampling is sensitive to the number of minority classes, and suffers from performance loss on majority classes. It suggests that a good solution should overcome the overfitting problem of oversampling but not by cutting down the size of majority classes.

Based on the first part of analysis, we have investigated the generalization ability of ensemble algorithms including AdaBoost.NC to deal with multi-class imbalance data, with the aim of tackling multi-class imbalance problems effectively. Extensive experiments are carried out on a set of benchmark data sets with multiple minority and/or majority classes. When the ensembles are trained without using class decomposition, our experimental results show that AdaBoost.NC working with random oversampling can produce better G-mean and minority-class recall than the others, which indicates good generalization for the minority class and the superior ability to balance the performance across different classes.

Our results also show that using class decomposition (the one-against-all scheme in our experiments – OAA) does not provide any advantages in multi-class imbalance learning. For AdaBoost.NC, its G-mean and minority-class recall is even weakened significantly by the use of class decomposition. The reason for this performance degradation seems to be the loss of global class distribution information in the process of class decomposition. An improved combination method for the OAA scheme is therefore proposed in this chapter, which assigns different weights to binary classifiers learnt from the sub-problems after the decomposition. The weights are decided by the proportion of the corresponding class within the data set, which delivers the distribution information of each class. By doing so, the effectiveness of AdaBoost.NC in G-mean and minority-class recall is improved significantly.

In regard to other methods, SMOTEBoost shows quite stable performance with good MAUC and G-mean in general. Oversampling itself does not bring much benefit to AdaBoost. Undersampling harms majority-class performance greatly.

Finally, we compare the ensembles without using OAA to the ones using OAA with the weighted combination method. The result suggests that it is not necessary to use class decomposition, and learning from the whole data set directly is sufficient for class imbalance learning techniques to achieve good performance.

Future work of this study includes: (a) an in-depth study of conditions, including parameter values, under which an ensemble approach, such as AdaBoost.NC, is able to improve the performance of multi-class imbalance problems; currently, the parameter of λ in AdaBoost.NC is pre-defined, and a large λ shows greater benefits; (b) investigation of other class imbalance learning methods into how their effectiveness is affected by multi-class; (c) a detailed analysis of why AdaBoost.NC seems to perform well in terms of G-mean and minority-class recall while SMOTEBoost is good at MAUC; (d) investigation of new ensemble algorithms that combine the strength of AdaBoost.NC and SMOTEBoost; (e) a theoretical framework for analyzing multi-class imbalance problems since it is unclear how an imbalance rate could be more appropriately defined.

We have ended our investigations concerning ensembles for class imbalance learning up to this point. The conclusions of this thesis and directions for the future work will be explained systematically in the following chapter.

CHAPTER 7

CONCLUSIONS AND FURTHER WORK

The primary questions that this thesis has answered are *"Is ensemble diversity helpful for class imbalance learning? How can we exploit diversity to improve the classification of class imbalance problems?"*. They are important questions for two reasons. First, many ensemble approaches have been proposed with empirical success in various class imbalance applications. It is useful to understand why they believe that ensemble learning is a good choice and the role of diversity in class imbalance learning as the most important feature of an ensemble model. Second, most existing ensemble solutions work on the data level by manipulating training data to correct the uneven class distribution or on the algorithm level by making models cost-sensitive, which often require a good understanding of data and careful parameter settings. It is meaningful to address class imbalance from the new angle of making use of diversity, to simplify the learning procedure and achieve better or at least competitive performance. This chapter summarizes the contributions of the thesis as the answers to the research questions stated in the Introduction chapter and gives directions for future work.

7.1 Conclusions

The focal point of this thesis is ensemble diversity for class imbalance learning. With this in mind, we investigated the theoretical and empirical role of ensemble diversity in dealing with class imbalance problems, and discovered that it has a significant and positive effect. To better consider the minority class of a class imbalance problem and better balance the overall performance by making use of diversity, we studied an important ensemble learning technique, negative correlation learning (NCL), since it encourages diversity explicitly during training. We proposed an ensemble algorithm for classification using the idea of NCL, named AdaBoost.NC, to overcome the problems of poor flexibility and efficiency of existing NCL algorithms. Its effectiveness in class imbalance learning was then investigated and exploited in both two-class and multi-class scenarios. The details of the contributions and significance of this thesis are explained as follows.

7.1.1 Role of Ensemble Diversity in Class Imbalance Learning

Ensemble learning has become a major technique of dealing with class imbalance problems. The interaction effect of ensemble members, termed diversity, is claimed to be helpful. However, no study has investigated the role of ensemble diversity in class imbalance learning before the thesis. Therefore, the thesis starts with the following questions stated in section 1.3.1: what is the relationship between ensemble diversity and the performance measures used in class imbalance learning? Is introducing diversity beneficial to the classification of the minority/majority classes in the presence of imbalanced data?

Chapter 3 gave a systematic and in-depth study of the relationship between ensemble diversity and generalization from both single-class and overall performance aspects (Wang and Yao, 2009a). To answer the first question, we derived mathematical links between single-class measures and diversity based on several classification patterns of ensembles. Diversity is measured by Q-statistic. The single-class measures include recall, precision and F-measure. We found that their changing behaviours along with Q-statistic are correlated to each other and fall into six possible situations. Based on the understanding of classification patterns and characteristics of classifying a class imbalance problem, we explained that diversity is expected to have a positive effect on the minority class and a negative effect on the majority class.

For the second question, we verified the results of how the single-class measures behave by varying the diversity degree of ensembles on a set of artificial and real-world imbalanced data sets. We investigated the impact of diversity based on correlation analysis with comprehensive experimental discussions. Strong correlations were found. Diversity showed a positive impact on the minority class in terms of recall and F-measure in general, which is achieved by making the ensemble produce broader and less overfitting classification boundaries for the minority class. A performance reduction on the majority class caused by diversity was observed in term of recall and F-measure on real-world data sets.

The relationship between diversity and overall performance, including G-mean and AUC, was studied empirically. Strong correlations existed. Diversity was shown to be beneficial to both measures.

Unlike the weak correlation found in the current literature between diversity and overall accuracy, we obtained strong correlations for single-class and overall performance measures in class imbalance scenarios. It is a meaningful result, suggesting the usefulness of diversity in this type of classification problems. The positive role of diversity suggests an alternative way of handling class imbalance problems, which can be used to develop better solutions. In addition, the results here help us to understand how the major performance measures are affected by ensemble diversity.

7.1.2 A New NCL Method for Classification

Aiming at a better solution for class imbalance problems by making use of ensemble diversity, chapter 4 focused on the ensemble learning technique of negative correlation learning (NCL), which considers the accuracy-diversity trade-off explicitly and is well-known for its good generalization ability. We pointed out two main issues of existing NCL algorithms for classification ensembles: their theoretical grounding is only valid in the regression context, not in classification; they require limited types of base learners and can be computationally expensive.

To overcome these problems relating to the research questions in section 1.3.3, we proposed a new ensemble algorithm, called AdaBoost.NC, combining the ideas of NCL and AdaBoost (Wang et al., 2010). It was shown to be more efficient than other NCL algorithms and exhibit better generalization performance than AdaBoost on general classification tasks. It allows wider choice of base learners. The "diversity-encouraging" term (i.e. ambiguity) used in AdaBoost.NC was obtained from the 0-1 error function. It describes the classification difference within the ensemble and provides theoretical support to adapt the traditional NCL based on the squared error function in regression for classification. AdaBoost.NC can be viewed as the first NCL algorithm specializing in classification problems.

As another contribution, the effectiveness of AdaBoost.NC was associated with error correlation of the ensemble in the sequential training context with theoretical and empirical explanations (Wang and Yao, 2010). We proved how reducing error correlation could benefit the classification accuracy. AdaBoost.NC was shown to produce smaller error correlation and thus a lower test error than AdaBoost with less overfitting classification boundaries. It explains how error correlation contributes to accuracy from a whole new angle.

7.1.3 How to Better Deal with Two-Class Imbalance Problems?

Based on the results from chapters 3 and 4, NCL algorithms including AdaBoost.NC were applied to tackling two-class imbalance problems in chapter 5. Because most existing ensemble methods in the literature suffer from overfitting and over-generalization problems depending on the selected training strategies and corresponding parameter settings, the study here aims at the following questions stated in section 1.3.2: if and how we can take advantage of ensemble diversity to better deal with class imbalance; if NCL methods including AdaBoost.NC can be good solutions to class imbalance problems. The compelling advantages of considering diversity in class imbalance learning are: no data information is lost; no data generation method is involved in training. It reduces the dependence of the algorithm on resampling techniques and training data. In addition, manipulating ensemble diversity is independent of the base learning algorithm, so it is free of choosing any base learner and more flexible than algorithm-level solutions.

AdaBoost.NC was studied thoroughly on both artificial and real-world imbalanced data, in order to understand the effect of its “diversity-encouraging” term in imbalanced scenarios and exploit this effect to improve the performance. Two other typical NCL algorithms (i.e. CELS and NCCD) and some state-of-art ensemble solutions for class imbalance problems were also investigated and compared through comprehensive experiments. The results suggested that NCL methods integrated with random oversampling are effective in finding minority class examples without losing the overall performance compared to other methods. This was achieved by providing broader and less overfitting classification boundaries for the minority class. The oversampling level and imbalance rate of training data were shown not to be crucial factors to influence the effectiveness. Particularly, AdaBoost.NC tree ensembles presented very promising generalization results in terms of AUC and minority-class performance.

The research here shows the usefulness of ensemble diversity in solving class imbalance problems, and opens up a novel way to deal with real-world applications that suffer from class imbalance difficulties. It is worth mentioning that our methods were also tested and presented good results on a set of software engineering tasks, which is a new application to this field and worthy of further investigation.

7.1.4 Challenges and Potential Solutions for Multi-Class Imbalance Problems

As an important extension to two-class cases, multi-class imbalance problems pose new challenges that have not drawn much attention. The ineffectiveness of two-class imbalance learning techniques caused by multi-class has been reported. In chapter 6, we aimed to find out what problems multi-class can cause and how it affects the classification performance in the presence of imbalanced data. Two types of multi-class imbalance problems, i.e. the multi-minority and multi-majority cases, were studied by applying random oversampling and undersampling techniques. Both types showed strong negative correlations with the performance measures, which implied that the performance decreases as the number of imbalanced classes increases. The multi-majority case was shown to be more harmful than the multi-minority case, because the imbalance rate became more severe. Regarding the class imbalance learning techniques, oversampling did not help the classification and caused overfitting; undersampling was sensitive to the number of minority classes and suffered from great performance loss on majority classes. This is the first systematic study of multi-class for class imbalance learning by providing separate and in-depth discussions of multi-minority and multi-majority cases. The results reveal possible issues that a class imbalance learning technique could confront when dealing with multi-class tasks, and provide guidance for designing better solutions.

Following the multi-class investigation and the promising results of AdaBoost.NC on two-class imbalance problems in chapter 5, chapter 6 continued the NCL study with the aim of tackling multi-class imbalance problems effectively and directly. AdaBoost.NC was applied to a set of benchmark data sets with multiple minority and/or majority classes. It was shown to have good generalization for the minority class and balance the performance across different classes well in terms of G-mean without applying any class decomposition, when working with random oversampling. This work provides a direct and effective way of dealing with multi-class imbalance problems from both minority-class and overall performance aspects.

7.2 Further Work

This section describes several directions for future research that may improve and extend the work in this thesis.

7.2.1 Sensitivity of Diversity Measures to Class Imbalance

In chapter 3, we studied the impact of diversity measured by Q-statistic on the classification performance of class imbalance problems. One reason for choosing Q-statistic is its less sensitivity to imbalanced distributions according to our empirical results and existing studies (Kuncheva and Whitaker, 2001, 2003). It is reflected in the observation that the members of an ensemble with overall high diversity tend to make diverse decisions on each class regardless of whether the class is minority or majority. For the consideration that there are other diversity measures available to assess the disagreement degree of classification ensembles, the work here arouses our interest in their sensitivity to class imbalance: which diversity measures are sensitive to class imbalance? Which ones are not? How do they behave in different imbalanced scenarios? Such investigation can help us to understand the role of diversity in class imbalance learning further and its relationship to overall and single-class performance from different perspectives.

7.2.2 Discrimination Between "Good" diversity and "Bad" Diversity

In the classification pattern analysis of chapter 3, we explained that diversity can be either beneficial or harmful to classification accuracy depending on which pattern the ensemble belongs to. We defined "good" diversity if it improves the performance; otherwise, it was treated as "bad" diversity. However, it is hard to judge whether good or not for a real-world problem in most cases. It would be very helpful to derive a measure or develop a method that can discriminate between "good" and "bad" diversity in the future, and guide the ensemble about when to consider diversity. We could start from the definitions of the classification patterns in this chapter, which have suggested some possible factors that the "quality" of diversity might rely on.

7.2.3 The AdaBoost.NC Algorithm

The ensemble algorithm proposed in chapter 4, AdaBoost.NC, employed a penalty term to balance the trade-off between accuracy and diversity explicitly and was shown to have good generalization performance. There is still a lot of room to further discuss and improve the algorithm:

1) The penalty currently utilizes the ambiguity term decomposed from the 0-1 error function to express the disagreement degree within the ensemble. In the future, we would like to consider other loss functions from which we may obtain different forms of ambiguity to substitute the one used in this algorithm. In addition, we found a direct relation of ambiguity to the diversity measure of entropy in section 4.2. In fact, we may consider other existing diversity measures to encourage diversity in AdaBoost.NC and see if and how its performance gets affected, although ambiguity has been shown to have stronger correlation with generalization error than others (Chen, 2008).

2) In section 4.3, we discussed how AdaBoost.NC's performance is affected by the parameter of penalty strength λ. We found that the best λ varies with the solving problems and base learning algorithms. The performance of AdaBoost.NC based on decision trees appeared to be more sensitive to λ than that based on neural networks. Besides, a large λ did not degrade the performance necessarily, depending on data domains. It would be very meaningful to give a specific range for setting λ in the future to guide the use of AdaBoost.NC. It would be also interesting to explore how it performs by using other types of base learners.

3) In section 4.3, we discussed the computational cost of AdaBoost.NC compared with other NCL algorithms, where AdaBoost.NC showed much less training time. Most of the data sets we used in the current study are quite small. We would like to consider real large data sets in the future. A theoretical complexity analysis for AdaBoost.NC's scalability would be also useful.

4) As an improved version of Boosting integrated with the idea of NCL, AdaBoost.NC was compared with the conventional AdaBoost and NCL algorithms in chapter 4. For

a further study and a complete understanding, we would like to compare AdaBoost.NC with other improved Boosting algorithms in the literature, such as BrownBoost (Freund, 2001), Modest AdaBoost (Vezhnevets and Vezhnevets, 2005), FloatBoost (Li and Zhang, 2004), etc., which have achieved varying degrees of success in specific problems.

7.2.4 Error Correlation in Sequential Training

In section 4.4, we proved how error correlation benefits classification accuracy in the sequential training context to explain the effectiveness of AdaBoost.NC. The main assumption of this proof is based on is that the individual classifier at the i-th training iteration is only correlated with its previous one built at the (i-1)-th iteration in making errors. It is based on the fact that the weights of training examples in AdaBoost are updated according to the accuracy of the current classifier rather than the ensemble. However, it is still not clear whether an indirect correlation exists between classifiers that are not next to each other through the example weights. In the future work, we would like to analyze this error correlation in depth between individual classifiers for the ensembles with a sequential training framework and study how it changes along with the training process. It may provide more information to explain the performance of AdaBoost and AdaBoost.NC in terms of error correlation.

7.2.5 Conditions of Using AdaBoost.NC in Class Imbalance Learning

In chapters 5 and 6, we investigated the generalization ability of AdaBoost.NC to solve class imbalance problems. We showed that AdaBoost.NC tree ensembles are good at identifying minority class examples and balancing the performance across classes when working with random oversampling. It is worthy of more discussions about the effectiveness of AdaBoost.NC with different settings, such as considering other base learning algorithms. Currently, the key parameter λ in AdaBoost.NC is pre-defined. A large λ is preferable. How large λ could be is another important issue to study in the future.

7.2.6 Analysis of Multi-Class Imbalance Learning

With respect to multi-class imbalance learning, we studied the classification difficulties caused by multi-class and potential solutions in chapter 6 of this thesis. The negative effects of multi-class were found and analyzed based on random oversampling and random undersampling. We would like to investigate how other class imbalance learning methods are affected by multi-class. For example, is there any method that is less sensitive to multi-class? Would they present similar behaviours to oversampling/undersampling, as the number of imbalanced classes increases?

Second, there is still a lack of theoretical framework to define this problem properly. For example, how should we define the imbalance rate when there exist multiple minority-majority pairs of classes? In two-class cases, it can be simply expressed by the proportion of the only minority class in the whole data set. In addition, not all minority classes suffer from recognition difficulties in some cases. How should we judge and handle this kind of situations?

Regarding the evaluation criteria for multi-class imbalance learning, we had some interesting observations from the experimental section in chapter 6. We noticed that an ensemble method with relatively high MAUC could be quite bad at recognizing examples of minority classes. It may imply a weak point of MAUC. Similarly, G-mean captures minority-class performance better, but may overlook the performance among majority classes. A better evaluation metric might be desirable that combines the strength of MAUC and G-mean. A related observation in the same section was that no single method produced consistently better G-mean and MAUC at the same time. We would like to find out the reasons and investigate new approaches that are good at both in the future.

7.2.7 Application of Class Imbalance Learning Methods to Software Engineering

In chapter 5, we solved a set of software engineering problems with the task of defect prediction by using class imbalance learning methods. All these data sets are very imbalanced, because defects are much less likely to occur than non-defects. Our method showed great potential of recognizing defects. In the field of software engineering, although some machine learning techniques were shown to be helpful for checking software quality automatically (Menzies et al., 2007, 2004), no prior work has considered class imbalance learning methods to facilitate this practical problem. In the future, we would like to explore the usefulness of class imbalance learning methods systematically for tackling this problem, in comparison with other representative methods adopted in the field of software engineering. Moreover, it would be interesting to utilize domain knowledge of software engineering, such as some data preprocessing and feature selection techniques, to further improve our methods.

APPENDIX A

CONNECTION BETWEEN AMBIGUITY AND BIAS-VARIANCE-COVARIANCE DECOMPOSITIONS FOR REGRESSION ENSEMBLES

The ambiguity decomposition (Krogh and Vedelsby, 1995) and the bias-variance-covariance decomposition (Ueda and Nakano, 1996) provide theoretical groundings for NCL algorithms in the regression context. In the ambiguity decomposition, the squared error of a regression ensemble with a linear combination rule is broken into two components

$$\left(\bar{f} - y\right)^2 = \sum_i \alpha_i \left(f_i - y\right)^2 - \sum_i \alpha_i \left(f_i - \bar{f}\right)^2 . \tag{A.1}$$

The first term on the right hand side is the average of individual errors. The second is referred to as "ambiguity", the amount of variability among the ensemble members. It guarantees that the ensemble squared error is always less than or equal to the average individual error for any data point.

With respect to all possible training data sets, the bias-variance-covariance decomposition shows the expected squared error of an ensemble $\bar{f}$ as follows

$$E\left\{\left(\bar{f} - y\right)^2\right\} = \overline{bias} + \frac{1}{L}\overline{var} + \left(1 - \frac{1}{L}\right)\overline{covar}. \tag{A.2}$$

It explains the role of diversity in generalization in terms of the covariance between learners. It can become negative. In order to improve generalization, it would be ideal to decrease the covariance without causing any increases of the average bias or variance terms.

Brown gave the exact link between these two decompositions to show what portions of the bias-variance-covariance decomposition correspond to the ambiguity term (Brown, 2004; Brown et al., 2005). By calculating the expectation of the ambiguity decomposition with uniformly weighted learners, we can obtain

$$E\left\{\frac{1}{L}\sum_i (f_i - y)^2\right\} = \overline{bias}^2 + \Omega \tag{A.3}$$

$$E\left\{\frac{1}{L}\sum_i \left(f_i - \bar{f}\right)^2\right\} = \Omega - \frac{1}{L}\overline{var} - \left(1 - \frac{1}{L}\right)\overline{covar}, \tag{A.4}$$

where Ω reflects the interaction between the two components of the ambiguity decomposition,

$$\Omega = \overline{var} + \frac{1}{L}\sum_i \left(E\left\{f_i\right\} - E\left\{\bar{f}\right\}\right)^2. \tag{A.5}$$

It is the average variance of the individual learners, plus the average squared deviation of the expectations of the individuals from the expectation of the ensemble. Due to the existence of Ω, we cannot maximize the ambiguity without affecting other parts of the error.

APPENDIX B

CHOICE OF CLASSIFIER WEIGHT α_i IN ADABOOST.NC

We show how we determine the form of α_i as the weight of each individual classifier in the AdaBoost.NC algorithm in this section. Assuming the notation in Table 4.1, we first derive the upper bound on the training error of $\bar{f}$. By unraveling the updating rule, we have

$$D_{L+1}\left(x_j\right) = \frac{\left(\prod_{i=1}^{L}\left(p_i\left(x_j\right)\right)^{\lambda}\right)\exp\left(-\sum_{i=1}^{L}\alpha_i\left[f_i\left(x_j\right)=y_j\right]\right)}{N\prod_{i=1}^{L}Z_i}, \tag{B.1}$$

with $\sum_{j=1}^{N} D_{L+1}\left(x_j\right) = 1$ holding. If $f_i\left(x_j\right) \neq y_j$, we have

$$\sum_{i=1}^{L}\alpha_i\left[f_i\left(x_j\right)=y_j\right] \leq \sum_{i=1}^{L}\alpha_i\left[f_i\left(x_j\right)\neq y_j\right]. \tag{B.2}$$

Thus, the training error is upper bounded by

$$\begin{aligned}
\frac{1}{N}\sum_{j}\left[\bar{f}\left(x_j\right)\neq y_j\right] &\leq \frac{1}{N}\sum_{j}\exp\left(\sum_{i=1}^{L}\alpha_i\left[f_i\left(x_j\right)\neq y_j\right]-\sum_{i=1}^{L}\alpha_i\left[f_i\left(x_j\right)=y_j\right]\right)\\
&\leq \frac{1}{N}\sum_{j}\exp\left(1-\sum_{i=1}^{L}\alpha_i\left[f_i\left(x_j\right)=y_j\right]\right)\\
&= e\left(\prod_{i=1}^{L}Z_i\right)\sum_{j}\frac{D_{L+1}\left(x_j\right)}{\prod_{i=1}^{L}\left(p_i\left(x_j\right)\right)^{\lambda}} \text{ (according to Eq. B.1).}
\end{aligned}$$

Since the penalty term p_i ranges in [1/2, 1], the bound can be further relaxed by

$$\begin{aligned}\frac{1}{N}\sum_{j}\left[\bar{f}\left(x_j\right)\neq y_j\right] &\leq e\left(\prod_{i=1}^{L}Z_i\right)2^{\lambda L}\sum_{j}D_{L+1}\left(x_j\right)\\ &= e\left(\prod_{i=1}^{L}Z_i\right)2^{\lambda L}.\end{aligned}$$

In order to minimize training error, the learning objective on each Boosting iteration is to find α_i so as to minimize Z_i. The penalty $\left(p_i\left(x_j\right)\right)^{\lambda}$ can be treated as a misclassification cost C_j associated with the example x_j. From this point of view, AdaBoost.NC has the cost-sensitive Boosting framework AdaC2 proposed in (Sun et al., 2007), in which α_i is uniquely selected as

$$\alpha_i=\frac{1}{2}\log\left(\frac{\sum_{j,y_j=f_i(x_j)}C_jD_i\left(x_j\right)}{\sum_{j,y_j\neq f_i(x_j)}C_jD_i\left(x_j\right)}\right) \tag{B.3}$$

to minimize the training error bound greedily. It leads to the choice of α_i for AdaBoost.NC given in Table 4.1.

APPENDIX C

PAIRED COMPARISON OF ALGORITHMS ON TWO-CLASS IMBALANCE PROBLEMS

This section shows all the results of paired comparison on fifteen real-world data sets between algorithms discussed in chapter 5. Three tables are included for comparing AUC, minority-class recall and minority-class precision respectively. The upper triangle of each table presents the results of NN-based models, and the lower triangle presents the results of tree-based models. Each entry is the amount of win-tie-lose of the method in a row compared with the method in a column.

Table C.1: Paired comparison of AUC based on T-test with confidence level of 0.95 on 15 data sets. The upper triangle shows the results of NN-based models, and the lower triangle shows the results of tree-based models. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrSg	OvSg	UnSg	OrAda	OvAda	UnAda	OrNC2
OrSg	-	4-8-3	2-9-4	7-2-6	5-3-7	5-5-5	6-3-6
OvSg	2-6-7	-	3-8-4	5-4-6	5-3-7	5-5-5	4-5-6
UnSg	10-3-2	11-4-0	-	6-4-5	6-4-5	3-9-3	6-2-7
OrAda	14-0-1	14-0-1	12-2-1	-	0-14-1	2-9-4	2-11-2
OvAda	14-0-1	14-0-1	12-2-1	0-14-1	-	2-11-2	4-10-1
UnAda	13-1-1	15-0-0	15-0-0	5-8-2	5-9-1	-	5-6-4
OrNC2	13-1-1	14-1-0	12-2-1	7-8-0	6-9-0	10-1-4	-
OrNC9	14-0-1	15-0-0	13-2-0	5-8-2	5-8-2	4-8-3	1-10-4
OvNC2	14-0-1	14-0-1	12-2-1	8-6-1	7-7-1	7-4-4	2-12-1
OvNC9	13-1-1	15-0-0	13-2-0	9-4-2	9-4-2	9-4-2	4-8-3
UnNC2	13-1-1	14-1-0	15-0-0	6-6-3	7-6-2	2-12-1	5-4-6
UnNC9	11-1-3	12-2-1	12-1-2	5-4-6	4-5-6	1-7-7	4-2-9
SMB	14-0-1	14-0-1	12-2-1	2-11-2	0-13-2	4-8-3	1-9-5
JSB	8-3-4	9-2-4	6-2-7	2-2-11	2-2-11	1-4-10	2-1-12
RAB	12-2-1	13-1-1	12-1-2	1-11-3	1-11-3	2-7-6	1-6-8

	OrNC9	OvNC2	OvNC9	UnNC2	UnNC9	OrCELS	OvCELS
OrSg	6-5-4	5-2-8	4-6-5	5-6-4	9-4-2	5-7-3	4-8-3
OvSg	5-7-3	2-6-7	3-7-5	6-3-6	8-3-4	5-6-4	5-6-4
UnSg	6-5-4	4-3-8	3-7-5	4-7-4	9-5-1	6-4-5	5-8-2
OrAda	5-10-0	3-4-8	6-3-6	7-4-4	8-4-3	6-2-7	9-2-4
OvAda	5-10-0	3-5-7	5-4-6	6-7-2	8-5-2	5-4-6	7-4-4
UnAda	4-8-3	3-5-7	4-8-3	4-10-1	8-7-0	6-3-6	5-8-2
OrNC2	5-9-1	0-10-5	6-3-6	5-6-4	7-4-4	5-4-6	6-3-6
OrNC9	-	0-7-8	3-6-6	3-9-3	5-7-3	5-3-7	5-3-7
OvNC2	6-8-1	-	6-8-1	7-6-2	8-6-1	7-3-5	6-5-4
OvNC9	6-9-0	4-8-3	-	5-7-3	8-6-1	6-5-4	4-7-4
UnNC2	4-8-3	3-6-6	1-7-7	-	5-9-1	6-2-7	2-9-4
UnNC9	4-5-6	3-4-8	0-6-9	0-7-8	-	6-2-7	1-8-6
OrCELS	-	-	-	-	-	-	3-6-6
SMB	3-9-3	1-8-6	2-6-7	3-7-5	7-5-3	-	-
JSB	2-1-12	2-1-12	2-1-12	1-2-12	3-3-9	-	-
RAB	2-6-7	1-5-9	2-3-10	2-5-8	6-2-7	-	-

	UnCELS	OrNCCD	OvNCCD	UnNCCD	SMB	JSB	RAB
OrSg	4-9-2	9-5-1	10-3-2	8-6-1	11-3-1	14-1-0	7-3-5
OvSg	5-7-3	8-4-3	9-3-3	8-4-3	7-7-1	14-1-0	6-5-4
UnSg	5-10-0	10-3-2	10-3-2	8-5-2	9-5-1	14-1-0	7-5-3
OrAda	6-3-6	8-3-4	9-4-2	7-5-3	10-4-1	13-1-1	2-13-0
OvAda	6-3-6	8-4-3	10-3-2	7-4-4	10-4-1	14-1-0	3-12-0
UnAda	5-6-4	8-4-3	10-3-2	8-4-3	8-6-1	15-0-0	6-8-1
OrNC2	7-2-6	7-2-6	9-3-3	7-3-5	8-3-4	12-2-1	3-8-4
OrNC9	5-4-6	6-5-4	10-1-4	6-4-5	7-4-4	12-2-1	3-8-4
OvNC2	8-2-5	8-4-3	10-3-2	8-2-5	11-2-2	14-1-0	10-3-2
OvNC9	6-5-4	8-5-2	10-2-3	7-3-5	10-4-1	14-0-1	7-3-5
UnNC2	6-4-5	7-4-4	9-3-3	7-4-4	7-5-3	14-1-0	5-5-5
UnNC9	3-5-7	7-5-3	9-3-3	4-5-6	5-4-6	15-0-0	3-5-7
OrCELS	2-8-5	6-7-2	10-2-3	4-7-4	7-4-4	13-1-1	7-3-5
OvCELS	3-9-3	7-7-1	10-4-1	5-8-2	7-8-0	15-0-0	5-5-5
UnCELS	-	8-5-2	10-3-2	4-10-1	7-4-4	14-1-0	6-4-5
OrNCCD	-	-	8-6-1	3-5-7	5-5-5	13-1-1	5-2-8
OvNCCD	-	-	-	2-5-8	3-3-9	10-4-1	2-3-10
UnNCCD	-	-	-	-	5-5-5	14-1-0	4-4-7
SMB	-	-	-	-	-	13-2-0	2-7-6
JSB	-	-	-	-	2-1-12	-	0-2-13
RAB	-	-	-	-	1-8-6	11-3-1	-

Table C.2: Paired comparison of recall of the minority class based on T-test with confidence level of 0.95 on 15 data sets. The upper triangle shows the results of NN-based models, and the lower triangle shows the results of tree-based models. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrSg	OvSg	UnSg	OrAda	OvAda	UnAda	OrNC2
OrSg	-	0-0-15	0-0-15	1-4-10	1-2-12	0-0-15	1-6-8
OvSg	9-6-0	-	1-4-10	12-3-0	8-7-0	1-2-12	13-1-1
UnSg	15-0-0	15-0-0	-	13-2-0	12-2-1	0-12-3	14-0-1
OrAda	2-12-1	0-5-10	0-1-14	-	1-6-8	0-0-15	3-8-4
OvAda	6-8-1	1-8-6	0-0-15	6-9-0	-	0-1-14	10-4-1
UnAda	15-0-0	15-0-0	3-12-0	15-0-0	15-0-0	-	14-1-0
OrNC2	0-8-7	0-5-10	0-0-15	0-8-7	0-4-11	0-0-15	-
OrNC9	0-2-13	0-1-14	0-0-15	0-2-13	0-1-14	0-0-15	0-4-11
OvNC2	7-7-1	2-4-9	0-0-15	5-10-0	0-11-4	0-0-15	12-3-0
OvNC9	14-1-0	13-1-1	1-5-9	14-1-0	13-2-0	0-4-11	15-0-0
UnNC2	15-0-0	15-0-0	5-10-0	15-0-0	15-0-0	4-11-0	15-0-0
UnNC9	15-0-0	15-0-0	4-11-0	14-1-0	15-0-0	6-8-1	15-0-0
SMB	11-3-1	5-8-2	0-1-14	14-1-0	6-8-1	0-0-15	13-2-0
JSB	2-8-5	1-4-10	0-2-13	2-7-6	1-4-10	0-1-14	3-8-4
RAB	1-10-4	0-4-11	0-0-15	0-13-2	0-5-10	0-0-15	6-8-1

	OrNC9	OvNC2	OvNC9	UnNC2	UnNC9	OrCELS	OvCELS
OrSg	7-4-4	0-2-13	0-1-14	0-0-15	0-0-15	3-6-6	0-1-14
OvSg	15-0-0	11-4-0	9-5-1	2-3-10	1-4-10	13-2-0	3-5-7
UnSg	15-0-0	13-2-0	13-2-0	3-7-5	1-12-2	14-1-0	8-2-5
OrAda	11-2-2	1-4-10	1-4-10	0-1-14	1-1-13	9-0-6	1-3-11
OvAda	15-0-0	5-7-3	3-4-8	1-3-11	1-2-12	10-4-1	1-5-9
UnAda	15-0-0	15-0-0	14-1-0	4-9-2	3-12-0	13-2-0	8-2-5
OrNC2	13-1-1	1-2-12	1-2-12	1-0-14	1-0-14	9-1-5	0-3-12
OrNC9	-	0-1-14	0-0-15	0-0-15	0-0-15	1-7-7	0-0-15
OvNC2	14-1-0	-	2-7-6	0-0-15	0-1-14	10-4-1	1-3-11
OvNC9	15-0-0	15-0-0	-	0-1-14	0-2-13	11-4-0	1-6-8
UnNC2	15-0-0	15-0-0	12-3-0	-	4-9-2	14-1-0	7-4-4
UnNC9	15-0-0	15-0-0	13-2-0	0-14-1	-	14-1-0	8-2-5
OrCELS	-	-	-	-	-	-	0-3-12
SMB	14-1-0	10-4-1	1-1-13	0-1-14	0-1-14	-	-
JSB	10-2-3	0-4-11	0-1-14	0-1-14	0-2-13	-	-
RAB	13-2-0	0-9-6	0-1-14	0-0-15	0-0-15	-	-

	UnCELS	OrNCCD	OvNCCD	UnNCCD	SMB	JSB	RAB
OrSg	0-0-15	5-7-3	2-3-10	0-1-14	1-6-8	5-6-4	1-4-10
OvSg	2-6-7	13-1-1	7-5-3	2-2-11	14-1-0	13-1-1	13-2-0
UnSg	6-5-4	14-1-0	10-2-3	5-3-7	15-0-0	15-0-0	14-1-0
OrAda	1-2-12	11-3-1	4-0-11	2-0-13	5-8-2	9-5-1	2-13-0
OvAda	1-4-10	13-1-1	6-4-5	2-0-13	9-5-1	11-3-1	9-5-1
UnAda	8-5-2	14-1-0	11-2-2	4-4-7	15-0-0	15-0-0	15-0-0
OrNC2	0-1-14	12-2-1	4-2-9	2-0-13	5-6-4	8-6-1	5-5-5
OrNC9	0-0-15	7-3-5	1-4-10	0-1-14	1-4-10	5-5-5	2-2-11
OvNC2	0-3-12	13-1-1	6-4-5	1-1-13	9-5-1	14-0-1	12-3-0
OvNC9	1-5-9	13-1-1	7-5-3	1-1-13	13-1-1	14-0-1	11-4-0
UnNC2	8-5-2	14-1-0	12-1-2	5-2-8	15-0-0	14-1-0	15-0-0
UnNC9	5-8-2	14-0-1	12-0-3	3-5-7	15-0-0	14-1-0	14-1-0
OrCELS	0-2-13	7-6-2	3-5-7	0-1-14	6-1-8	7-2-6	4-2-9
OvCELS	6-3-6	13-1-1	10-2-3	4-1-10	13-2-0	14-1-0	13-1-1
UnCELS	-	14-0-1	11-1-3	2-6-7	15-0-0	14-1-0	12-3-0
OrNCCD	-	-	1-5-9	0-0-15	2-3-10	4-4-7	1-3-11
OvNCCD	-	-	-	1-1-13	10-1-4	13-0-2	10-2-3
UnNCCD	-	-	-	-	13-1-1	13-2-0	14-0-1
SMB	-	-	-	-	-	7-6-2	3-6-6
JSB	-	-	-	-	1-2-12	-	1-5-9
RAB	-	-	-	-	0-2-13	6-8-1	-

Table C.3: Paired comparison of precision of the minority class based on T-test with confidence level of 0.95 on 15 data sets. The upper triangle shows the results of NN-based models, and the lower triangle shows the results of tree-based models. Each entry presents the amount of win-tie-lose of a method in a row comparing with the method in a column.

	OrSg	OvSg	UnSg	OrAda	OvAda	UnAda	OrNC2
OrSg	-	8-2-5	10-1-4	4-5-6	5-4-6	9-2-4	4-6-5
OvSg	1-7-7	-	10-5-0	1-6-8	0-10-5	13-2-0	2-5-8
UnSg	1-1-13	1-0-14	-	1-1-13	0-4-11	2-11-2	2-3-10
OrAda	7-7-1	9-6-0	13-1-1	-	4-10-1	13-1-1	4-11-0
OvAda	5-8-2	9-6-0	13-1-1	0-10-5	-	14-1-0	4-8-3
UnAda	1-2-12	1-1-13	9-6-0	1-1-13	1-1-13	-	1-3-11
OrNC2	9-5-1	10-4-1	12-1-2	6-8-1	7-7-1	12-1-2	-
OrNC9	4-4-7	4-3-8	6-2-7	4-2-9	3-4-8	5-3-7	1-6-8
OvNC2	7-6-2	9-6-0	13-1-1	1-14-0	6-9-0	12-2-1	1-9-5
OvNC9	2-6-7	5-7-3	13-1-1	1-5-9	2-6-7	13-1-1	2-3-10
UnNC2	1-3-11	1-3-11	8-7-0	1-1-13	1-1-13	4-10-1	2-1-12
UnNC9	1-2-12	1-3-11	6-5-4	1-1-13	1-1-13	5-5-5	2-1-12
SMB	4-10-1	10-5-0	14-0-1	0-9-6	3-10-2	14-0-1	1-6-8
JSB	6-5-4	7-4-4	9-2-4	1-7-7	4-5-6	9-2-4	2-5-8
RAB	5-7-3	8-6-1	12-2-1	0-13-2	4-10-1	11-3-1	1-7-7

	OrNC9	OvNC2	OvNC9	UnNC2	UnNC9	OrCELS	OvCELS
OrSg	6-7-2	5-4-6	7-3-5	9-1-5	10-0-5	5-5-5	6-4-5
OvSg	6-6-3	3-3-9	4-4-7	14-1-0	11-4-0	3-4-8	7-3-5
UnSg	5-5-5	2-2-11	1-5-9	7-3-5	7-6-2	2-5-8	5-2-8
OrAda	12-2-1	5-9-1	9-5-1	12-2-1	13-1-1	8-5-2	7-7-1
OvAda	9-4-2	5-7-3	7-6-2	14-1-0	15-0-0	6-4-5	8-6-1
UnAda	4-6-5	1-2-12	1-5-9	4-9-2	8-3-4	2-4-9	5-2-8
OrNC2	9-4-2	3-11-1	7-5-3	11-3-1	12-2-1	5-7-3	7-5-3
OrNC9	-	2-4-9	4-4-7	6-7-2	6-6-3	1-7-7	3-3-9
OvNC2	9-3-3	-	6-7-2	13-2-0	13-2-0	5-5-5	8-5-2
OvNC9	8-3-4	2-3-10	-	13-2-0	13-2-0	4-4-7	5-6-4
UnNC2	7-2-6	1-2-12	1-1-13	-	5-6-4	2-3-10	3-4-8
UnNC9	7-2-6	1-2-12	1-2-12	2-7-6	-	2-3-10	2-3-10
OrCELS	-	-	-	-	-	-	7-3-5
SMB	8-3-4	1-9-5	9-5-1	14-0-1	14-0-1	-	-
JSB	8-3-4	2-7-6	8-2-5	9-2-4	9-2-4	-	-
RAB	9-2-4	0-12-3	8-4-3	11-3-1	12-2-1	-	-

	UnCELS	OrNCCD	OvNCCD	UnNCCD	SMB	JSB	RAB
OrSg	8-2-5	7-7-1	7-5-3	10-2-3	6-6-3	4-6-5	5-4-6
OvSg	9-5-1	10-2-3	7-3-5	14-1-0	3-8-4	5-3-7	2-6-7
UnSg	6-4-5	9-3-3	5-4-6	9-5-1	2-3-10	5-3-7	2-3-10
OrAda	10-4-1	11-2-2	9-4-2	14-0-1	8-7-0	6-8-1	2-13-0
OvAda	9-6-0	11-3-1	7-5-3	15-0-0	6-8-1	6-4-5	3-9-3
UnAda	5-4-6	10-2-3	6-2-7	9-5-1	2-3-10	5-3-7	2-3-10
OrNC2	10-4-1	11-3-1	7-6-2	13-1-1	6-8-1	5-6-4	3-9-3
OrNC9	6-4-5	6-6-3	4-6-5	9-4-2	1-9-5	4-5-6	1-5-9
OvNC2	10-4-1	11-3-1	7-4-4	15-0-0	7-6-2	6-4-5	3-8-4
OvNC9	10-3-2	10-3-2	8-2-5	15-0-0	4-6-5	6-4-5	2-6-7
UnNC2	5-2-8	9-3-3	6-1-8	9-5-1	2-3-10	5-3-7	1-4-10
UnNC9	2-5-8	8-4-3	5-2-8	6-7-2	2-2-11	5-3-7	1-3-11
OrCELS	8-4-3	10-5-0	9-3-3	12-1-2	6-6-3	6-4-5	2-6-7
OvCELS	7-3-5	9-4-2	6-5-4	12-2-1	4-8-3	6-4-5	4-5-6
UnCELS	-	11-1-3	7-1-7	11-2-2	2-7-6	5-4-6	1-6-8
OrNCCD	-	-	4-5-6	5-4-6	3-3-9	2-4-9	2-2-11
OvNCCD	-	-	-	8-4-3	4-5-6	4-3-8	3-3-9
UnNCCD	-	-	-	-	1-2-12	4-2-9	1-2-12
SMB	-	-	-	-	-	4-5-6	1-7-7
JSB	-	-	-	-	4-5-6	-	1-9-5
RAB	-	-	-	-	5-7-3	6-8-1	-

LIST OF REFERENCES

Kazi Md. Rokibul Alam and Md. Monirul Islam. Combining boosting with negative correlation learning for training neural network ensembles. In *International Conference on Information and Communication Technology*, 2007.

Roberto Alejo, Jose M. Sotoca, R. M. Valdovinos, and Gustavo A. Casañ. The multi-class imbalance problem: Cost functions with modular and non-modular neural networks. In *The Sixth International Symposium on Neural Networks*, volume 56, pages 421–431, 2009.

Kamal M. Ali and Michael J. Pazzani. On the link between error correlation and error reduction in decision tree ensembles. Technical report ics-tr-95-38, University of California, Irvine, Department of Information and Computer Science, 1995.

R. Barandela, R.M. Valdovinos, and J.S. Sanchez. New applications of ensembles of classifiers. *Pattern Analysis and Applications*, 6(3):245–256, 2003.

Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *Special issue on learning from imbalanced datasets, Sigkdd Explorations*, 6(1):20–29, 2004.

Gustavo E.A.P.A. Batista, Ronaldo C. Prati, and Maria C. Monard. Balancing strategies and class overlapping. *Advances in Intelligent Data Analysis*, 3646:24–35, 2005.

Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–139, 1999.

Gary Boetticher, Tim Menzies, and Thomas J. Ostrand. Promise repository of empirical software engineering data, 2007. URL `http://promisedata.org/repository`.

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

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

Leo Breiman. Prediction games and arcing algorithms. *Neural Computation*, 11(7), 1999.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, October 2001.

Gavin Brown. *Diversity in Neural Network Ensembles.* PhD thesis, School of Computer Science, University of Birmingham, 2004.

Gavin Brown. Ensemble learning. *Encyclopedia of Machine Learning*, pages 1–24, 2010.

Gavin Brown and Xin Yao. On the effectiveness of negative correlation learning. In *Proceedings of first UK workshop on computational intelligence*, pages 57–62, 2001.

Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2004.

Gavin Brown, Jeremy L. Wyatt, and Peter Tino. Managing diversity in regression ensembles. *Journal of Machine Learning Research*, 6:1621–1650, 2005.

Philip K. Chan and Salvatore J. Stolfo. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *Knowledge Discovery and Data Mining*, pages 164–168, 1998.

Zeke S. H. Chan and Nik Kasabov. Fast neural network ensemble learning via negative-correlation data correction. *IEEE Transactions on Neural Networks*, 16(6):1707– 1710, 2005a.

Zeke S. H. Chan and Nikola Kasabov. A preliminary study on negative correlation learning via correlation-corrected data. *Neural Processing Letters*, 21(3):207–214, 2005b.

Nitesh V. Chawla. C4.5 and imbalanced data sets: Investigating the effect of sampling method, probabilistic estimate, and decision tree structure. In *Workshop on Learning from Imbalanced Datasets II, ICML, Washington DC, 2003.*, pages 1–8, 2003.

Nitesh V. Chawla and Jared Sylvester. Exploiting diversity in ensembles: Improving the performance on unbalanced datasets. *Multiple Classifier Systems*, 4472:397–406, 2007.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:341–378, 2002.

Nitesh V. Chawla, Aleksandar Lazarevic, Lawrence O. Hall, and Kevin W. Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *Knowledge Discovery in Databases: PKDD 2003*, volume 2838, pages 107–119, 2003.

Nitesh V. Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Editorial: Special issue on learning from imbalanced data sets. *SIGKDD Explor. Newsl.*, 6(1):1–6, 2004.

Huanhuan Chen. *Diversity and Regularization in Neural Network Ensembles (Chapter 3).* PhD thesis, School of Computer Science, University of Birmingham, 2008, Url: http://www.comp.leeds.ac.uk/scshc/, 2008.

Huanhuan Chen and Xin Yao. Regularized negative correlation learning for neural network ensembles. *IEEE Transactions on Neural Networks*, 20(12):1962–1979, 2009.

Ken Chen, Bao-Liang Lu, and James T. Kwok. Efficient classification of multi-label and imbalanced data using min-max modular classifiers. In *International Joint Conference on Neural Networks*, pages 1770–1775, 2006.

Kevin J. Cherkauer. Human expert-level performance on a scientific image analysis task by a system using combined artificial neural networks. In *Proceedings of AAAI-96 Workshop on Integrating Multiple Learned Models for Improving and Scaling Machine Learning Algorithms*, pages 15–21, 1996.

Yun-Sheng Chung, D. Frank Hsu, and Chuan Yi Tang. On the diversity-performance relationship for majority voting in classifier ensembles. In *MCS'07: Proceedings of the 7th international conference on Multiple classifier systems*, pages 407–420, 2007.

Padraig Cunningham. Overfitting and diversity in classification ensembles based on feature selection. Technical report, Computer Science Technical Report TCD-CS-2000-07, Department of Computer Science, Trinity College Dublin, 2000.

Pádraig Cunningham and John Carney. Diversity versus quality in classification ensembles based on feature selection. *Machine Learning: ECML 2000*, 1810:109–116, 2000.

Hai H. Dam, Hussein A. Abbass, Chris Lokan, and Xin Yao. Neural-based learning classifier systems. *IEEE Transactions on Knowledge and Data Engineering*, 2007.

Thomas G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, August 2000a.

Thomas G. Dietterich. Ensemble methods in machine learning. In *MCS '00: Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 1–15, London, UK, 2000b. Springer-Verlag. URL `citeseer.ist.psu.edu/dietterich00ensemble.html`.

Thomas G. Dietterich. Ensemble learning. *The Handbook of Brain Theory and Neural Networks*, 2002.

Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, pages 263–286, 1995.

Pedro Domingos and Michael Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29:103–130, 1997.

Chris Drummond and Robert C. Holte. Exploiting the cost (in)sensitivity of decision tree splitting criteria. In *Proceedings of the 7th International Conference on Machine Learning*, pages 239–246, 2000.

Chris Drummond and Robert C. Holte. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In *Proceedings of the International Conference on Machine Learning (ICML 2003) Workshop on Learning from Imbalanced Data Sets*, pages 1–8, 2003.

Charles Elkan. The foundations of cost-sensitive learning. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI'01)*, pages 973–978, 2001.

Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. In *Computational Intelligence 20*, volume 20, pages 18–36, 2004.

Kazuo J. Ezawa, Moninder Singh, and Steven W. Norton. Learning goal oriented bayesian networks for telecommunications risk management. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 139–147, 1996.

Wei Fan, Salvatore J. Stolfo, Junxin Zhang, and Philip K. Chan. Adacost: Misclassification cost-sensitive boosting. In *Proc. 16th International Conf. on Machine Learning*, pages 97–105, 1999.

Tom Fawcett. Roc graphs: Notes and practical considerations for researchers. *HP Labs, Palo Alto, CA, Technical Report HPL-2003-4*, 2003.

Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

Tom Fawcett and Foster Provost. Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 1(3):291–316, 1997.

Alberto Fernández, Mara José del Jesus, and Francisco Herrera. Multi-class imbalanced data-sets with linguistic fuzzy rule based classification systems based on pairwise learning. *Computational Intelligence for Knowledge-Based System Design*, 6178:89–98, 2010.

Joseph L. Fleiss. *Statistical methods for rates and proportions.* John Wiley and Sons, 1981.

A. Frank and A. Asuncion. UCI machine learning repository: http://archive.ics.uci.edu/ml, 2010.

Yoav Freund. An adaptive version of the boost by majority algorithm. *Machine Learning*, 43(3):293–318, 2001.

Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Proc. of the 13th. Int. Conf. on Machine Learning*, pages 148–156, 1996.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55: 119–139, 1997.

Giorgio Fumera and Fabio Roli. Linear combiners for classifier fusion: Some theoretical and experimental results. *Multiple Classifier Systems*, 2709:74–83, 2003.

Giorgio Fumera and Fabio Roli. A theoretical and experimental analysis of linear combiners for multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):942–956, 2005.

Nicolas Garcia-Pedrajas, Cesar Hervas-Martinez, and Domingo Ortiz-Boyer. Cooperative coevolution of artificial neural network ensembles for pattern classification. *IEEE Transactions on Evolutionary Computation*, 9(3):271– 302, 2005.

Giorgio Giacinto and Fabio Roli. Design of effective neural network ensembles for image classification purposes. *Image and Vision Computing*, 19(9-10):699–707, 2000.

Qiong Gu, Zhihua Cai, Li Zhu, and Bo Huang. Data mining on imbalanced data sets. In *International Conference on Advanced Computer Theory and Engineering 2008*, pages 1020–1024, 2008.

Hongyu Guo and Herna L. Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *SIGKDD Explor. Newsl.*, 6(1):30–39, 2004.

Xinjian Guo, Yilong Yin, Cailing Dong, Gongping Yang, and Guangtong Zhou. On the class imbalance problem. In *ICNC '08: Proceedings of the 2008 Fourth International Conference on Natural Computation*, pages 192–201, 2008.

Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *Advances in Intelligent Computing*, pages 878–887, 2005.

David J. Hand and Robert J. Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine Learning*, 45(2):171–186, 2001.

L. K. Hansen, C. Liisberg, and P. Salamon. Ensemble methods for handwritten digit recognition. In *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, pages 333–342, 1992.

Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, 1990.

Peter E. Hart. The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, 14(3):515–516, 1968.

Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. *The Annals of Statistics*, 26(2):451–471, 1998.

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

Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *IEEE International Joint Conference on Neural Networks*, pages 1322–1328, 2008.

Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, August 1998.

Tin Kam Ho, Jonathan J. Hull, and Sargur N. Srihari. Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):66–75, 1994.

Robert C. Holte, Liane E. Acker, and Bruce W. Porter. Concept learning and the problem of small disjuncts. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 813–818, 1989.

Fu Jie Huang, Zhihua Zhou, Hong-Jiang Zhang, and Tsuhan Chen. Pose invariant face recognition. In *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000*, pages 245–250, 2000.

Jason Van Hulse, Taghi M. Khoshgoftaar, and Amri Napolitano. Experimental perspectives on learning from imbalanced data. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 935–942, 2007.

Md. Monirul Islam, Xin Yao, and Kazuyuki Murase. A constructive algorithm for training cooperative neural network ensembles. In *IEEE Transactions on Neural Networks*, volume 14, page 4, 2003.

Md. Monirul Islam, Xin Yao, S. M. Shahriar Nirjon, Muhammad Asiful Islam, and Kazuyuki Murase. Bagging and boosting negatively correlated neural networks. *IEEE Transactions on Systems, Man and Cybernetics*, 38(3), 2008.

Nathalie Japkowicz. The class imbalance problem: Significance and strategies. In *Proceedings of the 2000 International Conference on Artificial Intelligence*, 2000a.

Nathalie Japkowicz. Learning from imbalanced data sets: A comparison of various strategies. In *AAAI Workshop on Learning from Imbalanced Data Sets*, pages 10–15, 2000b.

Nathalie Japkowicz. Concept-learning in the presence of between-class and within-class imbalances. *Lecture Notes in Computer Science*, 2056:67–77, 2001a.

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

Nathalie Japkowicz. Class imbalances: are we focusing on the right issue. In *Proceedings of the ICML'03 Workshop on Learning from Imbalanced Data Sets*, pages 17–23, 2003.

Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis*, 6(5):429 – 449, 2002.

Nathalie Japkowicz, Catherine Myers, and Mark A. Gluck. A novelty detection approach to classification. In *IJCAI*, pages 518–523, 1995.

Rong Jin and Jian Zhang. Multi-class learning by smoothed boosting. *Machine Learning*, 67(3):207–227, 2007.

Taeho Jo and Nathalie Japkowicz. Class imbalances versus small disjuncts. In *ACM SIGKDD Explorations Newsletter*, volume 6, pages 40–49, 2004.

Laurikkala Jorma. Improving identification of difficult small classes by balancing class distribution. In *Conference on artificial intelligence in medicine in Europe No8, Cascais , PORTUGAL*, volume 2101, pages 63–66, 2001.

Mahesh V. Joshi. On evaluating performance of classifiers for rare classes. *IEEE International Conference on Data Mining*, 0:641–661, 2002.

Mahesh V. Joshi, Vipin Kumar, and Ramesh C. Agarwal. Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *IBM Research Report*, pages 257–264, 2001.

Mahesh V. Joshi, Ramesh C. Agarwal, and Vipin Kumar. Predicting rare classes: can boosting make any weak learner strong? In *KDD '02: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 297–306, 2002.

Taghi M. Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Supervised neural network modeling: An empirical investigation into learning from imbalanced data with labeling errors. *IEEE Transactions on Neural Networks*, 21(5):813–830, 2010.

John Kirkland. Squad-based expert modules for closing diphthong recognition. In *Proceedings of the Second New Zealand International Two-Stream Conference on Neural Networks and Expert Systems*, pages 302–305, 1995.

Josef Kittler, Mohamad Hatef, Robert P.W. Duin, and Jiri Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

Ron Kohavi and David H. Wolpert. Bias plus variance decomposition for zero-one loss functions. *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 275–283, 1996.

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

Sotiris B. Kotsiantis, Dimitris Kanellopoulos, and Panayiotis Pintelas. Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, 30(1):25–36, 2006.

Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. In *Advances in Neural Information Processing Systems*, volume 7, pages 231–238, 1995.

Miroslav Kubat and Stan Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In *Proc. 14th International Conference on Machine Learning*, pages 179–186, 1997.

Miroslav Kubat, Robert Holte, and Stan Matwin. Learning when negative examples abound. In *The 9th European Conference on Machine Learning*, volume 1224, pages 146–153, 1997.

Matja Z. Kukar and Igor Kononenko. Cost-sensitive learning with neural networks. In *Proceedings of European Conference on Artificial Intelligence*, pages 445–449, 1998.

Ludmila I. Kuncheva. A theoretical study on six classifier fusion strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):281–286, 2002.

Ludmila I. Kuncheva. That elusive diversity in classifier ensembles. *Pattern Recognition and Image Analysis*, 2652:1126–1138, 2003.

Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms.* Wiley Press, 2004.

Ludmila I. Kuncheva and Christopher J. Whitaker. Ten measures of diversity in classifier ensembles: limits for two classifiers. In *A DERA/IEE Workshop on Intelligent Sensor Processing (Ref. No. 2001/050)*, pages 1–10, 2001.

Ludmila I. Kuncheva and Christopher J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51: 181–207, 2003.

Ludmila I. Kuncheva, Marina Skurichina, and Robert P.W. Duin. An experimental study on diversity for bagging and boosting with linear classifiers. *Information Fusion*, 3(4): 245–258, 2002.

Ludmila I. Kuncheva, Christopher J. Whitaker, Catherine A. Shipp, and Robert P.W. Duin. Limits on the majority vote accuracy in classifier fusion. *Pattern Analysis and Applications*, 6(1):22–31, 2003.

Hyoung-Joo Lee and Sungzoon Cho. The novelty detection approach for different degrees of class imbalance. *Lecture Notes in Computer Science Series as the Proceedings of International Conference on Neural Information Processing*, 4233:21–30, 2006.

Cen Li. Classifying imbalanced data using a bagging ensemble variation. In *ACM-SE 45: Proceedings of the 45th Annual Southeast Regional Conference*, pages 203–208, 2007.

Stan Z. Li and Zhenqiu Zhang. Floatboost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1112–1123, 2004.

T. Warren Liao. Classification of weld flaws with imbalanced class data. *Expert Systems with Applications*, 35(3):1041–1052, 2008.

Yi Lin, Yoonkyung Lee, and Grace Wahba. Support vector machines for classification in nonstandard situations. *Machine Learning*, 46(1-3):191–202, 2002.

Charles X. Ling, Jin Huang, and Harry Zhang. Auc: a statistically consistent and more discriminating measure than accuracy. In *Proceedings of 18th International Conference on Artificial Intelligence (IJCAI2003)*, pages 329–341, 2003.

Xu-Ying Liu and Zhi-Hua Zhou. The influence of class imbalance on cost-sensitive learning: An empirical study. In *Sixth IEEE International Conference on Data Mining (ICDM'06)*, pages 970–974, 2006.

Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class imbalance learning. *IEEE Transactions on Systems, Man and Cybernetics*, 39(2):539–550, 2009.

Yong Liu and Xin Yao. Negatively correlated neural networks can produce best ensembles. *Australian Journal of Intelligent Information Processing Systems*, pages 176–185, 1997.

Yong Liu and Xin Yao. Ensemble learning via negative correlation. *Neural Networks*, 12 (10):1399–1404, 1999a.

Yong Liu and Xin Yao. Simultaneous training of negatively correlated neural networks in an ensemble. *IEEE Transactions on Systems, Man and Cybernetics, PartB: Cybernetics*, 29(6):716–725, 1999b.

Yong Liu, Xin Yao, and Tetsuya Higuchi. Evolutionary ensembles with negative correlation learning. In *IEEE Transactions on Evolutionary Computation*, volume 4, pages 380– 387, 2000.

Marcus A. Maloof. Learning when data sets are imbalanced and when costs are unequal and unknown. In *ICML Workshop on Learning from Imbalanced Data Sets II, (2003)*, 2003.

Larry Manevitz and Malik Yousef. One-class document classification via neural networks. *Neurocomputing*, 70(7-9):1466–1481, 2007.

Larry M. Manevitz and Malik Yousef. One-class svms for document classification. *Journal of Machine Learning Research*, 2:139–154, 2001.

Dragos D. Margineantu and Thomas G. Dietterich. Pruning adaptive boosting. In *Proceedings of the 14th International Conference on Machine Learning*, pages 211–218, 1997.

David Mease, Abraham J. Wyner, and Andreas Buja. Boosted classification trees and class probability/quantile estimation. *The Journal of Machine Learning Research*, 8: 409–439, 2007.

Tim Menzies, Justin DiStefano, Andres Orrego, and Robert Chapman. Assessing predictors of software defects. In *Workshop on Predictive Software Models*, 2004.

Tim Menzies, Jeremy Greenwald, and Art Frank. Data mining static code attributes to learn defect predictors. *IEEE Transactions on Software Engineering*, 33(1):2–13, 2007.

Maria Carolina Monard and Gustavo E.A.P.A. Batista. Learning with skewed class distributions. In *Advances in Logic, Artificial Intelligence and Robotics*, pages 173–180, 2002.

Douglas C. Montgomery. *Design and Analysis of Experiments*. Great Britain: John Wiley and Sons, 6th edition, 2004.

Alejandro Murua. Upper bounds for error rates of linear combinations of classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):591–602, 2002.

David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

Guobin Ou and Yi Lu Murphey. Multi-class pattern classification using neural networks. *Pattern Recognition*, 40(1):4–18, 2007.

Derek Partridge and Wojtek Krzanowski. Software diversity: practical statistics for its measurement and exploitation. *Information and Software Technology*, 39(10):707–717, 1997.

Ronald K. Pearson, Gregory E. Gonye, and James S. Schwaber. Imbalanced clustering for microarray time-series. In *Proceedings of the ICML'03 Workshop on Learning from Imbalanced Data Sets*, 2003.

Michael P. Perrone and Leon N Cooper. When networks disagree: Ensemble methods for hybrid neural networks. *Artificial Neural Networks for Speech and Vision*, pages 126–142, 1993.

Charles A. Pierce, Richard A. Block, and Herman Aguinis. Cautionary note on reporting eta-squared values from multifactor anova designs. *Educational and Psychological Measurement*, 64:916–924, 2004.

Robi Polikar. Ensemble based systems in decision making. In *IEEE Circuits and Systems Magazine*, pages 21–45, 2006.

Ronaldo C. Prati, Gustavo E.A.P.A. Batista, and Maria Carolina Monard. Class imbalances versus class overlapping: An analysis of a learning system behavior. *Lecture Notes in Computer Science*, 2972:312–321, 2004.

Foster Provost. Machine learning from imbalanced data sets 101. In *Proceedings of the AAAI'00 Workshop on Imbalanced Data Sets*, 2000.

Foster Provost and Pedro Domingos. Well-trained pets: Improving probability estimation trees. In *CDER Working Paper: 00-04-IS, Stern School of Business, New York University.*, 2000.

Foster Provost and Tom Fawcett. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *Proceedings of the Third International Conference of Knowledge Discovery and Data Mining*, pages 43–48, 1997.

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

J. Ross Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, March 1986.

J. Ross Quinlan. *C4.5: Programs for Machine Learning.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

J. Ross Quinlan. Bagging, boosting, and c4.5. In *The 1996 13th National Conference on Artificial Intelligence, AAAI 96*, pages 725–730, 1996.

Nageswara S.V. Rao. On fusers that perform better than best sensor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(8):904–909, 2001.

Bhavani Raskutti and AdamKowalczyk. Extreme re-balancing for svms: a case study. *SIGKDD Explorations*, 6(1):60–69, 2004.

Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

C.J. Van Rijsbergen. *Information Retrieval.* London, Butterworths, 1979.

Brian D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge, UK: Cambridge University Press, 1996.

Robert E. Schapire. A brief introduction to boosting. In *Proceedings of the 16th international joint conference on Artificial intelligence*, pages 1401–1406, 1999.

Robert E. Schapire. The boosting approach to machine learning: An overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, pages 1–23, 2002.

Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26 (5):1651–1686, 1998.

Bernhard Scholkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

Chris Seiffert, Taghi M. Khoshgoftaar, and Jason Van Hulse. Hybrid sampling for imbalanced data. In *IEEE International Conference on Information Reuse and Integration*, pages 202–207, 2008a.

Chris Seiffert, Taghi M. Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Improving learner performance with data sampling and boosting. In *20th IEEE International Conference on Tools with Artificial Intelligence, 2008.*, pages 452–459, 2008b.

Catherine A. Shipp and Ludmila I. Kuncheva. Relationships between combination methods and measures of diversity in combining classifiers. *Information Fusion*, 3(2):135–148, 2002a.

Catherine A. Shipp and Ludmila I. Kuncheva. An investigation into how adaboost affects classier diversity. In *IPMU*, pages 203–208, 2002b.

Forrest Shull, Vic Basili, Barry Boehm, A. Winsor Brown, Patricia Costa, Mikael Lindvall, Dan Port, Ioana Rus, Roseanne Tesoriero, and Marvin Zelkowitz. What we have learned about fighting defects. In *Eighth IEEE Symposium on Software Metrics*, pages 249–258, 2002.

David B. Skalak. The sources of increased accuracy for two proposed boosting algorithms. In *American Association for Artificial Intelligence, AAAI-96, Integrating Multiple Learned Models Workshop*, pages 1–6, 1996.

Marina Skurichina, Liudmila I. Kuncheva, and Robert P.W. Duin. Bagging and boosting for the nearest mean classifier - effects of sample size on diversity and accuracy. *Multiple Classifier Systems*, pages 307–311, 2002.

Peter H.A. Sneath and Robert R. Sokal. *Numerical Taxonomy.* W. H. Freeman, London, UK, 1973.

Yanmin Sun, Andrew K.C. Wong, and Yang Wang. Parameter inference of cost-sensitive boosting algorithms. In *Proceedings of 4th International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 21–30, 2005.

Yanmin Sun, Mohamed S. Kamel, and Yang Wang. Boosting for learning multiple classes with imbalanced class distribution. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 592–602, 2006.

Yanmin Sun, Mohamed S. Kamel, Andrew K.C. Wong, and Yang Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378, 2007.

John A. Swets. Measuring the accuracy of diagnostic systems. *Science*, 240(4857):1285 – 1293, 1988.

Aik Choon Tan, David Gilbert, and Yves Deville. Multi-class protein fold classification using a new ensemble machine learning approach. In *Genome Informatics*, volume 14, pages 206–217, 2003.

Ke Tang, Ponnuthurai N. Suganthan, and Xin Yao. An analysis of diversity measures. *Machine Learning*, 65:247–271, 2006.

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

Kai Ming Ting. A comparative study of cost-sensitive boosting algorithms. In *Proc. 17th International Conf. on Machine Learning*, pages 983–990, 2000.

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

Iban Tomek. Two modifications of cnn. *IEEE Transactions on Systems, Man and Cybernetics*, 6(11):769–772, 1976.

Kagan Tumer and Joydeep Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection Science*, 8(3-4):385–403, 1996a.

Kagan Tumer and Joydeep Ghosh. Analysis of decision boundaries in linearly combined neural classifiers. *Pattern Recognition*, 29(2):341–348, 1996b.

Kagan Tumer and Joydeep Ghosh. Linear and order statistics combiners for pattern classification. *Combining Artificial Neural Networks*, pages 127–162, 1999.

Naonori Ueda and Ryohei Nakano. Generalization error of ensemble estimators. *IEEE International Conference on Neural Networks*, 1:90–95, 1996.

Rosa Maria Valdovinos and J. Salvador Sanchez. Class-dependant resampling for medical applications. In *Proceedings of the Fourth International Conference on Machine Learning and Applications (ICMLA'05)*, pages 351–356, 2005.

Hamed Valizadegan, Rong Jin, and Anil K. Jain. Semi-supervised boosting for multi-class classification. *Machine Learning and Knowledge Discovery in Databases*, 5212:522–537, 2008.

Alexander Vezhnevets and Vladimir Vezhnevets. 'modest adaboost'–teaching adaboost to generalize better. *Graphicon*, 2005.

Sofia Visa and Anca Ralescu. Issues in mining imbalanced data sets - a review paper. In *Proceedings of the Sixteen Midwest Artificial Intelligence and Cognitive Science Conference*, pages 67–73, 2005.

Benjamin X. Wang and Nathalie Japkowicz. Imbalanced data set learning with synthetic examples. In *IRIS Machine Learning Workshop*, 2004.

Shuo Wang and Xin Yao. Theoretical study of the relationship between diversity and single-class measures for class imbalance learning. In *IEEE International Conference on Data Mining Workshops*, pages 76–81, 2009a.

Shuo Wang and Xin Yao. Diversity analysis on imbalanced data sets by using ensemble models. In *IEEE Symposium on Computational Intelligence and Data Mining 2009*, pages 324–331, 2009b.

Shuo Wang and Xin Yao. The effectiveness of a new negative correlation learning algorithm for classification ensembles. In *IEEE International Conference on Data Mining Workshops*, pages 1013–1020, 2010.

Shuo Wang and Xin Yao. Negative correlation learning for class imbalance problems. *IEEE Transactions on Knowledge and Data Engineering*, page (under review), 2011.

Shuo Wang, Huanhuan Chen, and Xin Yao. Negative correlation learning for classification ensembles. In *International Joint Conference on Neural Networks, WCCI*, pages 2893–2900. IEEE Press, 2010.

Gary M. Weiss. Mining with rarity: a unifying framework. *SIGKDD Explor. Newsl.*, 6 (1):7–19, 2004.

Gary M. Weiss and Foster Provost. The effect of class distribution on classifier learning: An empirical study. *Technical Report ML-TR-43, Department of Computer Science, Rutgers University*, 2001.

Gary M. Weiss and Foster Provost. Learning when training data are costly: the effect of class distribution on tree induction. *Journal of Artificial Intelligence Research, 2003*, pages 315–354, 2003.

Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man and Cybernetics*, 2(3):408–421, 1972.

Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. San Francisco, CA: Morgan Kaufmann., 2nd edition, 2005.

Gang Wu and Edward Y. Chang. Class-boundary alignment for imbalanced dataset learning. In *Proceedings of the ICML 2003 Workshop: Learning with Imbalanced Data Sets II*, pages 49–56, 2003.

Rong Yan, Yan Liu, Rong Jin, and Alex Hauptmann. On predicting rare classes with svm ensembles in scene classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2003.

G. Udny Yule. On the association of attributes in statistics. *Philosophical Transactions of the Royal Society of London*, A194:257–319, 1900.

Bianca Zadrozny and Charles Elkan. Learning and making decisions when costs and probabilities are both unknown. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 204–213, 2001.

Jianping Zhang and Inderjeet Mani. knn approach to unbalanced data distributions: A case study involving information extraction. In *Workshop on Learning from Imbalanced Datasets II, ICML, Washington DC, 2003.*, pages 42–48, 2003.

Xing-Ming Zhao, Xin Li, Luonan Chen, and Kazuyuki Aihara. Protein classification with imbalanced data. *Proteins: Structure, Function, and Bioinformatics*, 70:1125–1132, 2008.

Zhi-Hua Zhou and Xu-Ying Liu. On multi-class cost-sensitive learning. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 567–572, 2006a.

Zhi-Hua Zhou and Xu-Ying Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. In *IEEE Transactions on Knowledge and Data Engineering*, volume 18, pages 63– 77, 2006b.

Xingquan Zhu. Lazy bagging for classifying imbalanced data. In *Seventh IEEE International Conference on Data Mining, 2007. ICDM 2007.*, pages 763–768, 2007.

Ling Zhuang and Honghua Dai. Parameter optimization of kernel-based one-class classifier on imbalance learning. *Journal of Computers*, 1(7):32–40, 2006.