KNOWLEDGE DISCOVERY FROM DATABASES:

COST-SENSITIVE AND IMBALANCE LEARNING

by

Zhuo Yang

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Business Administration

David Eccles School of Business

The University of Utah

December 2010

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Zhuo Yang**

has been approved by the following supervisory committee members:

Olivia R. Liu Sheng	, Chair	**9/24/2010** Date Approved
Rohit Aggarwal	, Member	**9/24/2010** Date Approved
Paul Hu	, Member	**9/24/2010** Date Approved
William Moore	, Member	**9/24/2010** Date Approved
Gautam Pant	, Member	**9/24/2010** Date Approved

and by **Olivia R. Liu Sheng** , Chair of

the Department of **Operations and Information Systems**

and by Charles A. Wight, Dean of The Graduate School.

ABSTRACT

In the current business world, data collection for business analysis is not difficult any more. The major concern faced by business managers is whether they can use data to build predictive models so as to provide accurate information for decision-making. Knowledge Discovery from Databases (KDD) provides us a guideline for collecting data through identifying knowledge inside data. As one of the KDD steps, the data mining method provides a systematic and intelligent approach to learning a large amount of data and is critical to the success of KDD. In the past several decades, many different data mining algorithms have been developed and can be categorized as classification, association rule, and clustering. These data mining algorithms have been demonstrated to be very effective in solving different business questions. Among these data mining types, classification is the most popular group and is widely used in all kinds of business areas. However, the exiting classification algorithm is designed to maximize the prediction accuracy given by the assumption of equal class distribution and equal error costs. This assumption seldom holds in the real world. Thus, it is necessary to extend the current classification so that it can deal with the data with the imbalanced distribution and unequal costs. In this dissertation, I propose an Iterative Cost-sensitive Naïve Bayes (ICSNB) method aimed at reducing overall misclassification cost regardless of class distribution. During each iteration, k nearest neighbors are identified and form a new training set, which is used to learn unsolved instances. Using the characteristics of the nearest neighbor method, I also develop a new under-sampling method to solve the imbalance problem in the second study. In the second study, I design a general method to deal with the imbalance problem and identify noisy instances from the data set to create a balanced data set for learning. Both of these two methods are validated using multiple real world data sets. The empirical results show the superior performance of my methods compared to some existing and popular methods.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

With the development of technology, people are equipped with more abilities to analyze large data sets for business decision-making. In the real world, most business problems can be divided into two categories: classifying an observation into one of several predefined groups and estimating the occurrence probability of certain events. To this end, different predictive models were developed and applied to solve these business questions. Among these methods, data mining techniques have been realized to be efficient and effective tools in building predictive models, especially for the classification method. As a nontrivial process of discovering implicit, useful, and comprehensive knowledge on large amounts of data, the performance of data mining techniques has been illustrated in different domains, such as engineering, marketing, science, etc. In this dissertation, I focus on addressing two specific applications of data mining techniques: cost-sensitive learning and imbalance learning that arise from the real business environments.

1.1 Cost-sensitive Learning

As one of the primary tasks of data mining, classification has been one of the popular research topics in many areas, such as machine learning, statistics, etc. Traditional classification algorithms focus on maximizing the overall accuracy or minimizing the error rate and are evaluated by the overall accuracy or error rate. Turney (2000) discussed that during the classification learning procedure, many costs occur, such as misclassification cost, data acquisition cost, learner cost, etc. People are becoming more and more interested in reducing the expected cost instead of improving the accuracy. Especially, for business professionals, they are actually more concerned with the misclassification cost or the business loss due to the inaccuracy of prediction models. However, existing classification algorithms, such as the Naïve Bayes and Decision Tree are designed on the assumption of equal error costs and equal class distributions. Given that the equal costs are hard to hold in the real world, it is necessary to incorporate unequal costs into the learning process and extend the current classification algorithms for cost-sensitive learning (Domingos 1999; Elkan 2001; Turney 1995; Zhou and Liu 2006).

Nowadays, many cost-sensitive classifiers have been developed, such as the cost-sensitive Neural Network (Kulan and Kononenko 1998; Zhou and Liu 2006) and the cost-sensitive Decision Tree (Drummond and Holte 2000; Erray and Hacid 2006), etc. There are also some general cost-sensitive learning methods including stratification (over-sampling and under-sampling) and threshold adjusting. In stratification, the change of training data distribution through removing instances or replicating existing instances will impact the prediction in accordance with the cost ratio. Instead of using 0.5 as the decision boundary, the adjusting method moves the decision threshold based on the given cost information. These general methods show some advantages in certain classifiers (like Decision Tree), but may not be appropriate to Naïve Bayes, which is very easy to use and has good classification performance as shown in previous studies. My study proposes a new Iterative Cost-sensitive Naïve Bayes (ICSNB) method. Applying the good performance of Naïve Bayes in the examples' ranks and incorporating the k nearest neighbor method, my approach has an even lower misclassification cost than existing cost-sensitive learning methods.

1.2 Imbalance Learning

As discussed above, the underlying assumption of different classification algorithms is equal class distributions and error costs. Equal class distributions do not hold in the real world either. It is very common that people are more interested in rare events. For example, compared to the entire population, the number of loyal customers for certain brands or stores is small (King and Zeng 2000). Another typical example is the medical diagnosis, where cancer patients account for only a small part in comparison to the healthy people. The data set consisting of unequal-sized classes is called the imbalanced data set and then learning on the imbalanced data set is the imbalance learning (Barandela et al. 2004; Guo and Vikto 2004; Japkowicz and Stephen 2002).

Unequal class distribution violates the basic assumption of traditional classification methods. When a data set is imbalanced, traditional predictive models and methods tend to favor the majority class, resulting in high overall accuracy, but detection rates with respect to the minority class often are not satisfactory. For instance, when a model is trained on a binary data set with 1% of its examples from the minority class, a 99% accuracy rate can be achieved by classifying all examples as belonging to the majority class. While a 99% accuracy is often considered excellent, such a model often has no practical value since our interests are often with the minority class. In the real world, people are frequently more interested in the minority class than in the majority class. Therefore, the existence of the imbalance problem in the real world and the target in imbalanced data force us to consider this issue when I build a classification model.

Another problem in building a predictive model is noisy data. Noisy data can cause a negative impact on the classification performance and degrade the generalizability of a prediction model. Such data usually cause the prediction on its neighbors far away from their true values. Therefore, identifying noisy instances and removing them from input will be crucial in building an accurate classification model. Accordingly, it is necessary to develop a new learning method, which is sensitive to imbalance, noise, and cost issues. In the second study of this dissertation, I focus on the under-sampling method to solve the imbalance problem in large-scale data sets. More specifically, I use the predicted misclassification cost as the measurement to identify and remove most costly samples from the majority group and then create a new and less-noisy training set for learning.

1.3 Overview of Dissertation

This dissertation aims at developing new approaches in cost-sensitive Naïve Bayes learning and imbalance learning. The structure of the dissertation consists of four chapters. Chapter 1 gives the overall introduction to cost-sensitive learning and the imbalance problem as well as a brief description of the proposed methods. Chapter 2 and Chapter 3 discuss the detailed method and experiment results as well as case studies. Chapter 4 summarizes the findings of the dissertation and discusses the directions for future work. More specifically, I highlight Chapter 2 and Chapter 3 as follows.

Section 2.1 introduces the basic concept about the Bayesian theory and its applications. I then summarize related prior research in Naïve Bayes, cost-sensitive learning, imbalance learning and bagging in Section 2.2. In Section 2.3, the theoretical foundation and proof of the proposed method are discussed. To demonstrate the performance of the new method, I use Sections 2.4, 2.5, and 2.6 to describe the empirical experiment and discuss the results. In the cost-based under-sampling study, the introduction and related literature are discussed in Section 3.1 and Section 3.2. Then, Section 3.3 explains the detailed procedure of the cost-based under-sampling method. Three real-world business cases are discussed in section 3.4. I further conduct some additional experiments to verify the consistent performance through varying the class distribution and cost ratios in Section 3.5. Finally, the conclusions and limitations are discussed in Section 3.6.

The main research question I explore is to develop some methods to solve cost-sensitive and imbalance problems in the real business world. As shown in Figures 1.1 and 1.2, at a high level, the first study iteratively generates a new training set to learn unsolved test samples, and the second study removes noisy samples from the training set to create a new balanced training set for learning. Both methods use the Nearest Neighbor theory during the learning procedure.

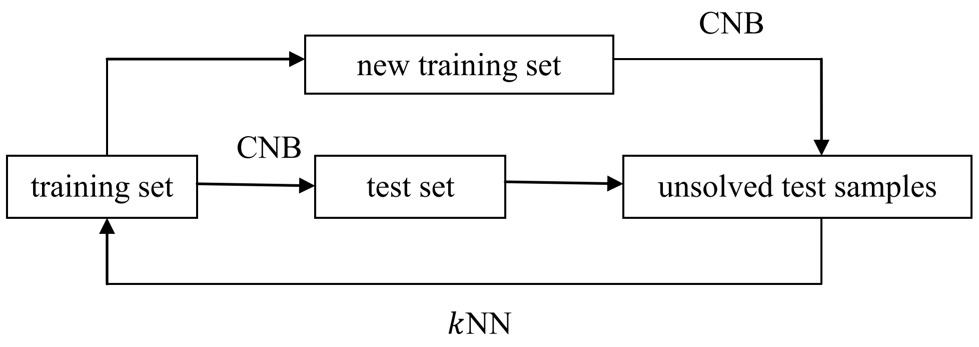


Figure 1.1: Process View of the Iterative Cost-sensitive Naïve Bayes Method (ICSNB)

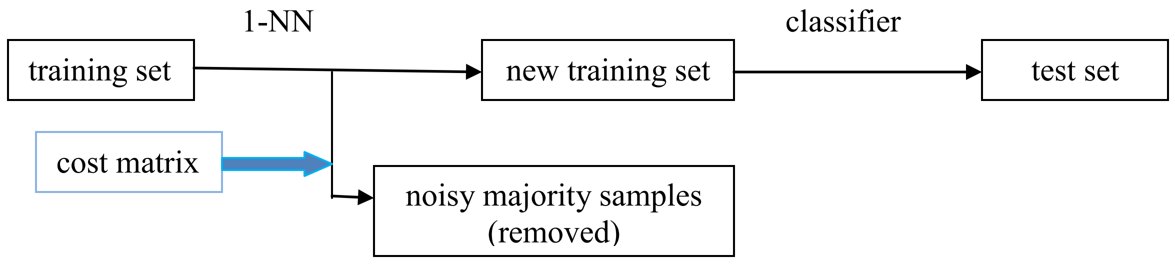


Figure 1.2: Process View of the Cost-based Under-sampling Method

CHAPTER 2

ITERATIVE KNN-BASED COST-SENSITIVE NAÏVE BAYES LEARNING

2.1 Introduction

There are many real-world problems requiring people to answer questions, like whether a certain event will happen and how likely it is. Especially in the business world, people are interested in predicting whether a customer will return or how likely he or she will spend more than one hundred dollars. The more accurate this type of analysis is, the more profit business organizations can make on those target customers. To this purpose, the classification method has been developed to build predict models helping people make better decisions. The classification learning is a very basic task in the area of data analysis and pattern recognition, which requires the construction of a classifier and assigns a class label to instances described by a set of attributes. Many classification algorithms have been developed in the past few decades, such as Decision Tree, Naïve Bayes, Neural Networks, SVM, etc.

Bayesian networks are often used for classification problems, in which a learner attempts to construct a classifier from a given set of training examples with class labels. Among different Bayesian network algorithms, Naïve Bayes is one of the most simple and effective inductive learning algorithms for machine learning and data mining. Researchers have found that Naïve Bayes is easy to use and quite robust in different domains (Friedman et al. 1997; Viaene et al. 2004). It is especially appropriate when the feature space is high (Hastie et al. 2009).

Furthermore, applying the Bayesian theory to solve real-world business problems has been showing great advantages in different fields, such as marketing and Information Systems. As discussed in the study of Rossi and Allenby (2003), Bayesian decision theory is ideally suited for many marketing problems and has been widely used in different ways. The performance is excellent when it is applied to solving a range of marketing problems from new product introduction to pricing. For instance, Scott and Yalch (1980) used a Bayesian analysis model to predict consumer response to an initial product trial and found that the Bayesian model was quite a useful framework for investigating the acceptance of new information as a result of the consumer's attribution process. Venkatesan et al. (2007) proposed a Bayesian decision theory-based selection strategy for identifying profitable customers. This Bayesian decision theory-based model provided better prediction accuracy than the benchmark model. The Bayesian method is also very easy to use with other methods so as to have even better performance than when used alone. Jen et al. (2003) proposed a model that combined the Bayesian method with the Poisson likelihood model. The model is used to predict the purchase frequency in the direct marketing and the performance demonstrates the advantage of using the Bayesian method relative to other approaches.

The Bayesian model has also received much attention in the Information Systems field. Sarkar and Siram (2001) developed a Bayesian model for early warning of bank failures. In their study, interrelated variables were grouped as one variable to tackle the violation of independent assumption for input attributes. Also, simple Naïve Bayes is very popular in the machine learning or other IT related areas.

Seewald (2007) compared the performance of simple Naïve Bayes and two extended variants in spam filtering. The result showed that there was not much difference among these three methods and verified the excellent performance of simple Naïve Bayes.

Although the excellent performance of the Bayesian model has been observed in many studies, most previous research focuses on improving prediction accuracy. The misclassification cost is another important factor or measurement that should be considered in the Bayesian model. Since uniform cost is not true in most business problems, for example, the benefit of correctly targeting a profitable customer is obviously higher than the average campaign cost. Therefore, people are more interested in how to minimize the cost in the Bayesian learning. This direction has drawn attention in machine learning research and has led to an increased interest for developing cost-sensitive classification methods (Turney 1997). Currently, there are only some general cost-sensitive methods that can be applied on Naïve Bayes, such as stratification (under-sampling and over-sampling), threshold adjusting, MetaCost (Domingos 1999), etc. But all these methods rely on probability estimations to make predictions. The major problem in the Naïve Bayes method is the unrealistic assumption of attributes independence, which causes high estimation bias. That the probability estimate in Naïve Bayes is unreliable has been verified theoretically and empirically (Friedman 1997; Bennett 2000; Frank et al. 2001). Although having the poor probability estimate, Naïve Bayes has a very strong feature in instances ranking that can tolerate the estimation error of class probabilities to some extent (Domingos and Pazzani 1997; Zhang and Su 2008). Using this feature, I develop a new iterative cost-sensitive Naïve Bayes method. The empirical experiments show the excellent performance in reducing misclassification cost compared to some existing and popular cost-sensitive methods.

2.2 Literature Review

2.2.1 Naïve Bayes

Naïve Bayes applies the Bayes rule in the prediction of classification problems. When applying the Bayes rule, the Naïve Bayes method will compute the probabilities of a given instance belonging to different classes and then assign the class label with the highest probability. Considering a simple classification problem, let X be a randomly selected sample from a data set and Y class label with k possible values. Estimating the probability (i.e., $p(Y = y_k|X)$) can help us predict which class X is more likely belong to through $\arg\max p(Y = y_k|X)$. Applying the Bayes rule, we can have

$$\hat{P}(Y = y_k|X) = \frac{\mathrm{P(X|Y=y_k)P(Y=y_k)}}{\mathrm{P(X)}} \qquad (2.1)$$

For the consideration of convenient computation on $\mathrm{P(X|Y = y_k)}$, the Naïve Bayes classifier has a very fundamental assumption, i.e., attributes conditional independence (Duda and Hart 1973; Good 1965). This assumption considers each input attribute is independent from one another given the class label. For instance, there are three attributes A, B, and C for the given data set. If we say A is independent of B and C, it means

$$P(A|B,C) = P(A|C) \qquad (2.2)$$

Or, as shown in Figure 2.1, there are only relationships between class label attribute C and input attributes ($A_1, A_2, A_3,\ and\ A_4$) if we assume all input attributes are independent.

Given this assumption, then for the equation (2.1), if X consists of n independent attributes, the probability of a class label value y_k for an unlabelled instance X is given by

$$\hat{P}(Y = y_k | X_1 \dots X_n) = \frac{P(Y=y_k) \prod P(X_i|Y=y_k)}{\sum_j P(Y=y_j) \prod_j P(X_i|Y=y_j)} \quad (2.3)$$

$$\propto P(Y = y_k) \cdot \prod P(X_i|Y = y_k)$$

If we have discrete value for the inputs, we can easily calculate the probabilities of each component of equation (2.3) through counting the number of instances of input containing specific attribute values. However, there is one danger of this counting method if there is no occurrence for certain attribute value in the training data, but in the testing data. If we still follow equation (2.3) to compute the probability, we may have zero estimation. To avoid that, there is a common method called "smoothing," which adds in some values to the denominator and the numerator. Provost and Domingos (2000) suggest using the Laplace smoothing. It is given by

$$\hat{P}(Y = y_k | X_1 \dots X_n) = \frac{P(Y=y_k) \prod P(X_i|Y=y_k) + l}{\sum_j P(Y=y_j) \prod_j P(X_i|Y=y_j) + lJ} \quad (2.4)$$

where J is the number of distinct values of what X_i can take on, and l determines the strength of this smoothing. Usually, l is set to 1. For the two-class problem, 2 is selected for J. Then, the Laplace smoothing method adjusts probability estimates to be closer to 0.5, which is actually not reasonable for an imbalance situation.

Another smoothing method is called m-estimation, which is given by

$$\hat{P}(Y = y_k | X_1 \dots X_n) = \frac{p(Y=y_k) \prod p(X_i | Y=y_k) + b.m}{\sum_j p(Y=y_j) \prod_j p(X_i | Y=y_j) + m} \qquad (2.5)$$

where b is the ratio of the target class and m is a parameter that controls how many scores are shifted towards b. Zadrozny and Elkan (2001) chose $b.m = 10$ in their paper and achieved a better performance.

Although the assumption of conditional independence is almost always violated in the real world, practical comparisons have often shown that Naïve Bayes performs surprisingly well. For example, in a study of head injury problem, Titterington et al. (1981) found that the independence model yielded the overall best result. Similarly, Mani et al. (1997) found that the independence Bayes model did best in a study comparing classifiers for predicting breast cancer recurrence. Besides studies in medicine, other research (Cestnik et al. 1987; Cestnik 1990; Pazzani et al. 1996; Frieman et al. 1997; Domingos and Pazzani 1997) also indicated that the independence Bayes model performed very well, often better than the alternatives.

Research has shown that the good performance of Naïve Bayes in prediction is not from its accurate probability estimation, but from the accurate rank of testing instances. And actually Naïve Bayes has a worse performance in predicting probability. Given the accurate estimation in rank closing to real rank, Naïve Bayes can do well in classification in terms of accuracy if we can find the right decision boundary.

2.2.2 Imbalance Problem in Classification

As I discussed above, Naïve Bayes does a good job in the ranking of testing instance. If we can find the optimal decision threshold/boundary, we still can have good prediction performance without accurate probability estimation. However, traditional Naïve Bayes chooses 0.5 as a decision threshold. For the two-class problem, it assumes the data set has an equal number of instances for each class. Actually, it is also an underlying assumption for different other classification algorithms. However, it is never true for any data set collected from the real world. When we want to predict a purchase decision from an individual level for existing customers, one inevitable problem is that customers with repeated purchase behavior are rare events for the whole customer base (King and Zeng 2000). Another typical example is the medical diagnosis domain, where sick patients are always a small group when compared to the entire population. We call this group of questions a class imbalance problem, and the data set is called imbalanced data, where the size of one class overwhelms that of the other class (Barandela et al. 2004; Guo and Vikto 2004; Japkowicz and Stephen 2002). This violates the basic assumption for applying traditional classification methods. When a data set is imbalanced, traditional predictive models and methods, such as classification methods, tend to favor the majority class, resulting in high overall accuracy, but detection rates with respect to the minority class often are not satisfactory. For instance, when a model is trained on a binary data set with 1% of its examples from the minority class, a 99% accuracy rate can be achieved by classifying all examples as belonging to the majority class. While 99% accuracy is often considered excellent, such a model often has no practical value since our interests are often on the minority class. In the real world, there are many examples where people are more interested in the minority class, not the majority class. This is also the reason why Laplace smoothing is not appropriate in correcting the Naïve Bayes probability estimation.

To solve imbalance problems in building predictive models using traditional classifiers, the common method is to change the data distribution and then create a balanced data set, called stratification or resampling. Changing data distribution can be done from two directions: under-samplng and over-sampling. The under-sampling method removes majority instances from a data set until the sizes of the two groups are equal, while the over-sampling method generates more minority instances and keeps the majority group unchanged.

2.2.3 Cost-sensitive Naïve Bayes

Inductive learning techniques, such as Naïve Bayes and Decision Tree, have met great success in building classification models. However, many previous research studies have only focused on how to minimize the classification errors (Mitchell 1997; Quinlan 1993). People have found that minimizing the classification cost attracts more and more attention instead of classification accuracy. For example, a marketing manager is more interested in how much profit loss it might bring caused by a predictive model misclassifying returning customers as being leaving customers. To this end, researchers developed different cost-sensitive learning methods trying to minimize the overall misclassification cost.

Cost-sensitive learning considers a variety of costs in various components and processes of learning (Turney 2000), with the goal of minimizing the costs individually or for the total cost. It is one of the most active and important research areas in data mining and machine learning, and it plays an important role in real-world data mining and machine learning applications. Turney (2000) provided a comprehensive survey of a large variety of different types of costs in data mining and machine learning, including misclassification costs, data acquisition costs (instance costs, attribute costs, and labeling costs), computation cost, human-computer interaction cost, and so on. The two most important types of costs are identified as misclassification costs and data acquisition costs. Misclassification costs are an extension of error rate as different types of errors (such as false positive and false negative) can have different costs. Data acquisition costs reflect how expensive it is to acquire extra information for assisting classification or building more accurate learning models (Weiss and Provost 2003; Yang et al. 2006; Zhu and Wu 2005). More and more research has been devoted to misclassification cost (Domingos 1999; Turney 1995; Zhou and Liu 2006).

Without loss of generality, in this study I assume binary classification: positive/1 and negative/0, and people are more interested in a positive class. In cost-sensitive learning, a cost matrix must be given. Table 2.1 is an example of a cost matrix. The first number in the parenthesis represents the predict value and the second number is the actual value. Thus, the same numbers mean correct prediction, otherwise, wrong prediction.

Usually, the costs of misclassifying a positive example and a negative example are not equal. If we are interested in positive class (for instance, loyal customers, patients with lung cancer), then C(0,1) will have higher value than C(1,0). In Naïve Bayes, once we get the probability of an instance belonging to different class, then this instance should be classified into the class that has the minimum expected cost. For an instance x, the optimal prediction is class 1 if and only if the expected cost of this prediction is less than or equal to the expected cost of predicting cost 0, i.e., if and only if

$$P(0|x)C(1,0) + P(1|x)C(1,1) \leq P(0|x)C(0,0) + P(1|x)C(0,1) \quad (2.6)$$

which is equivalent to

$$(1 - P)C(1,0) + P^*C(1,1) \leq (1-P)C(0,0) + P^*C(0,1) \quad (2.7)$$

where $P = P(1|x)$

Therefore, the optimal threshold is P^* such that

$$P^* = \frac{C(1,0)}{C(1,0)+C(0,1)} \quad (2.8)$$

if we set $C(0,0)$ and $C(1,1)$ to be zero.

Different cost-sensitive learning methods have been developed to extend existing classifiers, such as a cost-sensitive Neural Network (Kulan and Kononenko 1998; Zhou and Liu 2006), a cost-sensitive Decision Tree (Drummond and Holte 2000; Erray and Hacid 2006), etc.

Once cost matrix is given, P^* can then be calculated and fixed. It is very intuitive to think about whether we can make $\bar{y}$ approach to P^* through altering the distribution of the training data set. This is the method I discussed above, called cost-based resampling or stratification. Data resampling is a very common method in cost-sensitive learning (Elkan 2001; Weiss et al. 2007; Zadrozny et al. 2005; Zhou and Liu 2006), in which data distribution is altered artificially to reach the optimal threshold P^*. To change class distribution, we have two options as mentioned in the previous section, under-sampling or over-sampling, i.e., removing instances from the majority class or replicating instances for the minority class. Elakan (2001) gives us a general formula for how to do sampling correctly. In his formula, the number of negative (majority) examples in the training set should be multiplied by

$$\frac{P^*}{1-P^*}\frac{1-\bar{y}}{\bar{y}} \qquad (2.9)$$

In a special case, where $\bar{y} = 0.5$, equation (2.9) tells us that we need resample $\frac{C(1,0)}{C(0,1)} * N_0$ negative examples to have a new training set so that it matches the optimal decision threshold P^*.

Besides data sampling, another very popular and general cost-sensitive method is MetaCost (Domingos 1999). Considering the shortcomings of data sampling, information loss and overfitting, Domingos (1999) proposed the MetaCost method to make classification algorithms cost-sensitive without using sampling. He argued that the training examples should be relabeled with their optimal classes according to the cost matrix. Meanwhile, MetaCost, bagging (discussed next), is used as the ensemble method to learn class probability so as to improve probability accuracy.

However, either for data sampling or MetaCost, once Naïve Bayes is used to compute class probability, we do not have enough confidence about the estimation accuracy caused by the intrinsic high bias associated with Naïve Bayes. Using equation (2.6) may not give us an accurate prediction on class label.

2.2.4 Bagging

Methods for voting classification algorithms, such as bagging and boosting, have been shown to be very successful in improving the accuracy of certain classifiers. Voting algorithms can be divided into two types: those that adaptively change the distribution of the training set based on the performance of precious classifiers (boosting) and those that do not (bagging). In bagging, multiple versions of training data will be generated and used to have an aggregated classifier. Specifically, given a set D, of d tuples, bagging works as follows. For iteration i ($i = 1,2, \dots, k$), a training set, D_i of d tuples is sampled with replacement from the original set of tuples, D. Each training set is a bootstrap sample. Because sampling with a replacement is used, some of the original tuples of D may not be included in D_i, whereas others may occur more than once. A classifier model, M_i, is learned for each training set, D_i. To classify an unknown tuple, X, each classifier, M_i, returns its class prediction, which counts as one vote. The bagged classifier counts the votes and assigns the class with the most votes to X.

Breiman (1996a) indicated that a critical factor in whether bagging can improve accuracy is the stability of the procedure for constructing classifier M_i and bagging works well for unstable procedures. Meanwhile, Breiman (1996b) pointed out that most of classification models were unstable, such as Decision Tree, Neural Network, Naïve Bayes, except for k nearest neighbor. Especially for probability estimation, the evidence (Breiman 1996a) indicated that bagged estimates were likely to be more accurate than the single estimates.

Not only in the machine learning area, but also in marketing research, the importance of the bagging method in improving the prediction performance has been realized. In the context of marketing research, Lemmens and Croux (2006) draw attention to the competitive performance of bagging, an easy-to-use procedure, by repeatedly estimating a classifier to bootstrapped versions of the calibration sample. The result shows a significant increase using different evaluation measurements compared to the single prediction.

2.3 Proposed Method

Prior research has shown the high bias of probability estimation by the Naïve Bayes method empirically and theoretically (Friedman 1997; Bennett 2000; Frank et al. 2001). That is, probabilities estimated using the Naïve Bayes method are usually far away from actual probabilities, primarily due to the method's "Naïve" assumption of attribute independence (Friedman 1997). However, poor probability estimations may not lead to poor classification decisions, as illustrated in Figure 2.2. Let $P(x)$ be the actual probability of instance x belonging to the positive class and P^* be the optimal decision threshold. By (2.8), the optimal decision is to classify x as positive since $P(x) > P^*$. Two probability estimations of instance x belonging to the positive class, $\hat{P}(x)$ and $P'(x)$, are shown in Figure 2.2. It is clear that $\hat{P}(x)$ is a much more biased probability estimation than $P'(x)$. However, the classification decision based on $\hat{P}(x)$ (which is positive because $\hat{P}(x) > P^*$) coincides with the optimal decision, while the classification decision based on $P'(x)$ (which is negative because $P'(x) < P^*$) does not.

The example in Figure 2.2 shows that poor probability estimation does not mean poor classification decision; sometimes the contrary may be true. While a poor method for probability estimation, the Naïve Bayes method could be an ideal tool for classification. Indeed, prior research has empirically shown the superior classification performance of the Naïve Bayes method, when compared with other effective classification methods such as C4.5 (Langley et al. 1992; Domingos and Pazzani 1997). Friedman (1997) theoretically explains the superior classification performance of the Naïve Bayes method. Friedman (1997) shows that as long as the prior probability π of the positive class equals the optimal decision threshold P^* the Naïve Bayes method is the ideal choice for classification because the over-smoothing nature of the Naïve Bayes method causes probability estimations by the Naïve Bayes method shrinking toward π. In particular, the result of applying the Naïve Bayes method can be modeled as (Friedman 1997)

$$\hat{P}(x) = \left(1 - \alpha(x)\right)P(x) + \alpha(x) * \pi, \qquad (2.10)$$

where $\hat{P}(x)$ and $P(x)$ are the estimated and the actual probability of instance x belonging to the positive class, respectively, and $\alpha(x)$ is the over-smoothing coefficient, $0 \le \alpha(x) \le 1$.

While the Naïve Bayes method is ideal for classification when $\pi = P^*$, the condition seldom holds in the cost-sensitive environment. An effective cost-sensitive Naïve Bayes method needs to be developed for $\pi \neq P^*$. Using the over-smoothing characteristic of the Naïve Bayes method, I first analyze some interesting properties regarding $\hat{P}(x)$, the Naïve Bayes estimated probability of instance x belonging to the positive class.

Lemma 1: Given $\pi > P^*$,

(a) if $\hat{P}(x) \le P^*$, we have $P(x) < P^*$;

(b) if $\hat{P}(x) \ge \pi$, we have $P(x) > P^*$.

Proof. Let us first prove (a). By (9), $\hat{P}(x) \le P^*$ means

$$\left(1 - \alpha(x)\right)P(x) + \alpha(x) * \pi \le P^* .$$

Given $\pi > P^*$, we have

$$\left(1 - \alpha(x)\right)P(x) + \alpha(x) * P^* < P^*.$$

That is,

$$\left(1 - \alpha(x)\right)P(x) < (1 - \alpha(x))P^*.$$

Hence, $P(x) < P^*$.

We then prove (b). By (9), $\hat{P}(x) \geq \pi$ means

$$\big(1-\alpha(x)\big)P(x) + \alpha(x) * \pi \geq \pi.$$

The above inequality implies $(x) \geq \pi$. Given $\pi > P^*$, we have $P(x) > P^*$. This completes the proof.

By Lemma 1, given $\pi > P^*$, if the Naïve Bayes estimated probability $\hat{P}(x)$ is less than or equal to the optimal threshold P^*, the actual probability $P(x)$ is less than P^* and the instance *x* is labeled as negative according to (2.8). On the other hand, if $\hat{P}(x)$ is greater than or equal to the prior probability π of the positive class, the actual probability $P(x)$ is greater than P^* and the instance *x* is labeled as positive according to (2.8). As shown in Figure 2.3, using Lemma 1, I can determine labels for instances if their Naïve Bayes estimated probabilities $\hat{P}(x)$ are above (or on) the line of π or below (or on) the line of P^*. However, for instances with $\hat{P}(x)$ falling between P^* and π, their labels cannot be determined by Lemma 1.

Lemma 2: Given $\pi < P^*$,

(a) if $\hat{P}(x) \leq \pi$, we have $P(x) < P^*$,

(b) if $\hat{P}(x) \geq P^*$,we have $P(x) > P^*$.

Proof. We first prove (a). By (9), $\hat{P}(x) \leq \pi$ indicates

$$\big(1-\alpha(x)\big)P(x) + \alpha(x) * \pi \leq \pi. \qquad .$$

The above inequality implies that $(x) \leq \pi$. Given $\pi < P^*$, we have $P(x) < P^*$.

We prove (b) next. By (9), $\hat{P}(x) \geq P^*$ means

$$\big(1-\alpha(x)\big)P(x) + \alpha(x) * \pi \geq P^* .$$

Given $\pi < P^*$, we have

$$\big(1-\alpha(x)\big)P(x) > P^* - \alpha(x) * P^* .$$

Hence $P(x) > P^*$. This completes the proof.

By Lemma 2, given $\pi < P^*$, if the Naïve Bayes estimated probability $\hat{P}(x)$ is less than or equal to the prior probability π of the positive class, the actual probability $P(x)$ is less than P^* and the instance x is labeled as negative according to (2.8). On the other hand, if $\hat{P}(x)$ is greater than or equal to the optimal threshold P^*, the actual probability $P(x)$ is greater than P^* and the instance x is labeled as positive according to (2.8). As shown in Figure 2.4, using Lemma 2, I can determine labels for instances if their Naïve Bayes estimated probabilities $\hat{P}(x)$ are above (or on) the line of P^* or below (or on) the line of π. However, for instances with $\hat{P}(x)$ falling between π and P^*, their labels cannot be determined by Lemma 2.

Let U be the set of instances that cannot be labeled by applying Lemmas 1 and 2, which consists of instances with $\hat{P}(x)$ falling between π and P^*. These unlabeled instances need to be further learned with an appropriate training data set. I construct the training data set for U using the nearest neighbor approach (Cover and Hart 1967). The choice of the nearest neighbor approach is based on the fact that half information of an instance is contained in its nearest neighbor (Cover and Hart 1967). Further, prior research has shown that training data sets constructed using the nearest neighbor approach are effective for Naiva Bayes learning (Frank et al. 2003). Specifically, for each instance in U, its k nearest neighbors in the original training data set are identified. And the training data set for U consists of the k nearest neighbors of each instance in U. When identifying nearest neighbors, the distance between two instances is measured using the Euclidean distance (Frank et al. 2003; Han and Kamber 2005). The Euclidean distance $d(u, t)$ between an unlabeled instance x in U and an instance t in the original training data set is measured as

$$d(x, t) = \sqrt{(x_1 - t_1)^2 + (x_2 - t_2)^2 + \cdots + (x_n - t_n)^2} \qquad (2.11)$$

where n is the number of attributes of an instance. The k nearest neighbors of an unlabeled instance consist of its top- k closest instances in the original training data set. Once the training data set for U is constructed, I can apply the Naïve Bayes method to estimate probability $\hat{P}(x)$ for each instance x in U and employ Lemmas 1 and 2 to label instances in U. It is possible that there are still unlabeled instances left after the process. Therefore, I repeat the procedure of constructing training data for unlabelled instances and then labeling these instances using Lemmas 1 and 2 until there are no unlabeled instances left or there are some hard instances left that can never be labeled using the procedure. If there are hard instances left, I label them according to their probabilities $\hat{P}(x)$ estimated during the final run of the procedure. In particular, a hard instance is labeled as positive if $\hat{P}(x) \geq P^*$ and negative otherwise.

Based on the above discussion, I propose the Iterative Cost-sensitive Naïve Bayes (ICSNB) algorithm shown in Figure 2.5. The ICSNB algorithm takes the training data set L and the test data set T as inputs and assigns labels for each instance in T. The algorithm first employs the Naïve Bayes method to estimate probability $\hat{P}(x)$ for each instance x in T. It then labels instance x according to $\hat{P}(x)$ and Lemmas 1 and 2. Instances that cannot be labeled by Lemmas 1 and 2 are added to U, the set of unlabelled instances. The nearest neighbor procedure, NN(), is invoked next to construct the training data set for U and the algorithm calls itself recursively. The ICSNB algorithm terminates if there is no unlabelled instance (i.e., $U = \emptyset$) or there are only hard instances left. As shown in Figure 2.5, $|U| = |T|$ indicates that none of the instances in the test data set can be labeled by Lemmas 1 and 2. Hence, all instances in the test data set are added to U and they are all hard instances. The hard instances are finally labeled using procedure Hard().

2.4 Data Sets and Empirical Experiment Design

In this section, I describe how the experiment is designed to test my proposed iterative cost-sensitive Naïve Bayes method as well as how I select data sets and do data preprocessing or transformation. All the data sets used in the empirical study are from UCI Machine Learning Repository (Blake et al. 1999). In order to test the generality of my proposed method, I used several criteria in choosing experiment data sets. Table 2.2 gives the description for all the data sets used in the experiment. From Table 2.2, I can see that the data sets used in my empirical study have the following characteristics,

- Data size ranges from small to large. The smallest data set is *ionosphere* with only 351 data points, while the largest one (*magic gamma telescope*) has 19,020 data points. I did not choose data sets that are too large for computation convenience.
- Varied data dimensionality (number of attributes) from 6 attributes in *car evaluation* to 36 attributes in *chess.*
- Two-class data sets and multiclass data sets: the first five data sets are two-class data sets and the remaining three data sets have multivalues class label attributes.
- The selected data sets cover different domains, such as business (car evaluation), gaming (chess and tic-tac-toe), science (magic gamma telescope, ionosphere), etc.
- Data sets with continuous attributes only, categorical attributes only, or mixed.

For all these data sets, I have done preprocessing to clean the data before I put them into my experiment. As shown in Table 2.2, some data sets contain missing values. Missing values need to be taken care of before any analysis. In my experiment, I chose the common approach to handling missing values and all the missing values in the data set are replaced by mean (for continuous attribute) or mode (for categorical attribute) using Weka filter function. Then, I transfer all continuous attributes to categorical attributes with ten bins using the Weka discretize function and also let Weka decide the optimal bins. After discretization, some attributes only contained one value across the entire data set. If all instances in a data set only had one possible value for certain attributes, these attributes will have no impact on computing posterior probability. Therefore, I removed all attributes with only one value after the discretization operation. If you compare the information in Table 2.2 to the raw data in UCI database repository, you will find some data sets have fewer attributes.

In this study, I focused on binary or two-class problems. Some of the data sets used in the experiment are originally multiclass data sets and the class label attribute has more than two unique values. I converted multiclass data sets into two-class data sets through taking the interesting class as one class or target class and all other classes as the second class. This is a very convenient approach for solving the multiclass classification problem. Specifically, in data set *image segmentation*, I chosen “brickfact” as target (class 1) and the instances in the remaining classes are combined as one class (class 0). Similarly, I selected class 2 as the target class in data set *annealing*. But, in data set *car evaluation*, all acceptable cars including good and very good cars are used as target class.

Cost-sensitive learning is to minimize the overall misclassification cost for the given cost matrix. However, the UCI database repository does not provide any cost information for all eight data sets in my experiment. Therefore, I need to generate a synthetic cost matrix using certain mechanisms to compare cost reduction among different cost-sensitive learning methods. Prior research (Zhou and Liu 2006;), chosen randomly, value from uniform distribution between 1 and 10.

$$1 \leq C(i,j) \leq 10 \text{ for all } i \neq j \qquad (2.12)$$

where i is the predict class and j is the actual label. And, I assume correct prediction will not cause any cost, i.e., $C(0,0) = C(1,1) = 0$. Following the same procedure, I generate five cost matrices as shown in Table 2.3. In this study, I always take the interesting or target class as class 1 and the other class as class 0, and I also assume misclassifying an instance in class will cost more than that in class 0, i.e., $C(1,0) < C(0,1)$.

Using a variety of learning methods, I conducted a traditional evaluation through random selection 2/3 of the entire instances for training and the remaining one-third data for the test (Domingos 1999; Abe et al. 2004). I repeated this procedure twenty times and the results were the average of twenty such runs. I used overall misclassification cost as a measurement to evaluate model performance.

2.5 Experiment Results and Analysis

In the experiment, the benchmark methods used to compare with my proposed method are Cost-sensitive Naïve Bayes (CNB), Random Under-sampling (RUS), Random Over-sampling (ROS), and MetaCost. For Cost-sensitive Naïve Bayes, I still use traditional Naïve Bayes to compute the probability, but consider the cost information when I predict class label. Equation (2.7) is used to help make decision. For Random Under-sampling method, I keep target class (class 1) unchanged and randomly remove instances from class 0 following the formula (2.9). In Random

Over-sampling, formula (2.9) is used to calculate how many more instances need to be replicated in class 1. I apply Naïve Bayes in MetaCost and follow the same procedure (Domingos 1999) to do 50 times 100% bagging.

Table 2.4 is the summary of the overall comparison performance for all the data sets measured by misclassification cost through averaging costs over five cost matrices. The results show that the proposed method (ICSNB) outperforms all other benchmark methods. Comparing to other methods aiming to solve the cost-sensitive learning problem, my method can averagely reduce misclassification cost by 7.45% for CNB, 28.39% for RUS, 21.55% for ROS, and 44.89% for MetaCost, respectively. Not only on the two-class data sets, but also on multiclass data sets, the proposed method has a great advantage in reducing misclassification cost. As I describe before, I convert multiclass data sets into two-class data sets. Doing this transformation actually will increase the prediction difficulty since I combine multiple classes into one class. The involved instances may have totally different characteristics and some may be more close to the target instances. But, I artificially assign the same label to them and this may cause ever higher bias. It explains why the improvement is more significant for two-class data sets than for multiclass data sets between CNB and proposed method. Secondly, either for balanced or imbalanced data sets, my method shows consistent performance

For data set *ionosphere*, I observe a slight improvement compared to CNB, not like other two-class data sets. It also happens on the multiclass data set *image segmentation* and *annealing*. So, I take a further look and try to find out what the possible underlying reasons causing insignificant improvement are. My method is actually trying to improve the prediction accuracy for those instances falling into the middle area. If the improvement is not significant, it means either proposed method cannot give better prediction or there is no or very few instances for us to predict. After checking the performance of middle steps, I find that after the first cut using original π and P^*, there are only very few instances left in the middle area and leave us a very small space for the proposed method to improve the performance. Even I can make 100% prediction accuracy and still cannot reduce cost too much since the majority of input examples have been judged by traditional CNB.

Table 2.4 also exhibits that CNB is apparently better than ROS, RUS, and MetaCost except on the data set *magic gamma telescope*. RUS and ROS get similar performance and do not make too much difference for all eight data sets. Surprisingly, MetaCost almost has the worst performance among all benchmark methods except on data sets *tic-tact-toe* and *car evaluation*. Therefore, I can say that in existing cost-sensitive Naïve Bayes methods, traditional CNB usually can have better and consistent performance.

Table 2.5 and Table 2.6 give the detailed performance information for each individual cost matrix and corresponding graphs are shown in Figure 2.8 and Figure 2.9. These two tables help us further investigate the performance of the proposed method and the possible reasons explain the better performance in terms of misclassification cost reduction.

I first look at the performance between CNB and the proposed method. Under what conditions does the proposed method outperform CNB? Or, when is the performance of the proposed method close to that of CNB? I find that on data sets *chess* and *mushroom*, the proposed method is just slightly better than CNB under cost matrix 4 and 5. Similarly, I observe that under cost matrix 3, the proposed method has the lowest improvement on data set *tic-tac-toe*. If I calculate P^* from the given cost matrix, I will see that when P^* is very close to original π and difference between P^*

and π is very small, the improvement will be small. Different from the previous reason that only very few instances left in the middle area is caused by the data set itself, manually changing P^* and making it approach π can explain this insignificant performance.

Table 2.6 shows that my method even has more consistent performance on multiclass data sets than on two-class ones. Another interesting finding is that the MetaCost method does not display any benefit on cost-sensitive learning for all data sets and every cost matrix. This is because probability used in MetaCost to relabel class label is computed by Naïve Bayes. As I discuss before, probability estimation using Naïve Bayes is unreliable and with high bias. Given this high bias probability, the relabeling result is also questionable.

2.6 Conclusion and Discussion

In this dissertation, I analyze past works on cost-sensitive learning and Naïve Bayes classifier. Based on these works, I identified the research gap between existing cost-sensitive learning methods and Naïve Bayes classifier. Then, a new cost-sensitive method, iterative cost-sensitive Naïve Bayes (ICSNB), is proposed. My method applies the characteristics of Naïve Bayes classifier, takes iterative learning approach to improve prediction accuracy and thus reduces the overall misclassification cost. The performance of the proposed method is tested on eight UCI data sets covering different domains. I compare my method with other popular cost-sensitive learning methods, like data resampling and MetaCost.

The results suggest that the cost-sensitive method is easy and effective in reducing misclassification cost and has the consistent better performance across all the experimental data sets than the noncost-sensitive method. Especially, my proposed method reaches the best performance among all cost-sensitive methods. Traditional cost-sensitive Naïve Bayes is in the second position through adjusting decision threshold so as to make the decision cost-sensitive. Meanwhile, CNB is a conservative method that rarely causes negative effect and is easier to use than other methods. Sampling methods, either random under-sampling or random over-sampling, are found to be ineffective in improving prediction performance compared to CNB. MetaCost is significantly worse than others. Different from the study of Domingos (1999) in which MetaCost is applied to Decision Tree, I combine MetaCost with Naïve Bayes and results indicate that MetaCost is not a general method that can be applied to different classification algorithms.

In my proposed method, I do not specify how to find K to form a new training set. However, K is a very critical factor influencing the classification performance. The main difference between my method and common Naïve Bayes method is that I use KNN to have a new training set to learn the left over data samples. Therefore, choosing different Ks may have significantly different results that sometimes can be even worse than using Naïve Bayes only. Actually, in both case study and additional experiment, I did a greedy search between 1 and 100 to find an optimal K so as to have the lowest overall cost.

According to the finding across ten UCI data sets, the optimal K is usually between 10 and 30 for most of data sets, except for data set *magic gamma telescope*, in which final K is 65. I found that when K increases to over 50, the total misclassification cost can be even higher than that of smaller K less than 20. This observation is different from prior research (Frank et al. 2003) in that the KNN method is said not to be particularly sensitive to the choice of K as long as K is not too small and actually larger K is preferred. KNN theory also discusses that the performance of a classifier can be improved under larger K if given the infinite number of samples. However, I think in the real world K should not be selected as large as possible given the observations from my study. Choosing a large value of K may introduce noise into the new training set. I do not recommend using a large K, especially when the size of a given data set is small. Otherwise, it is very likely to still have the exact same training set as the original one and give the same result as using traditional Naïve Bayes method. Furthermore, it has been proved that the nearest neighbor (K=1) usually contains half the classification information (Cover and Hart 1967). So, small K may be enough to provide us information on estimating the probability of a given target sample. In conclusion, I think there are several issues that deserve my attention and can help us find optimal K with more efficient ways. First, I need to check how many instances are left after the first iteration. If the size of original training set is M and the remainder is m_1, then K should not be greater than M/m_1. Second, data sets with high-dimensionality should consider smaller K than ones with low-dimensionality. For low-dimensionality data sets, most of K's nearest neighbors may have similar distance to the target instance given large K. Hence, this may impose negative impact on classification performance.

In summary, this empirical study suggests that

1. The proposed iterative cost-sensitive Naïve Bayes (ICSNB) method has better performance in cost reduction than other cost-sensitive methods.
2. ICSNB can be used in two-class and multiclass data sets, and transformation is needed to multiclass the data sets. The effect of cost reduction is similar on two-class data set and multiclass data set.
3. Sampling methods do not show any advantage in Naïve Bayes learning.

4. MetaCost is found to be with the worst performance among all cost-sensitive methods in this study.
5. The choice of optimal *k* does not need to be very large. But I should consider relevant information, such as data size, class distribution, etc.

2.7 Limitations and Future Directions

This study also has its limitations that can be addressed by future studies. Considering the computational complexity, in this study, I discretize all continuous attributes. The key part of this study is to generate a new training set and therefore, the performance is obviously dominated by the distance measurement. As I discussed in previous sections, it is common to find more than *k* instances having the same distance to the target instance. Here, I just randomly select exactly K instances to form the new training set. Improvement can be made by using the actual attribute values. Also coming from the attribute transformation procedure, the probability estimation can be biased. I believe that there could be some potential space for performance improvement if continuous inputs are taken carefully using the Gaussian Naïve Bayes method.

Exploring general *k* Nearest Neighbor literature will be another direction for future work. In this study, optimal is found by greedy search. If I can discover certain relationships between optimal K and data size, attribute size, or distribution, etc, it will reduce much computation work.

Table 2.1: Example of Cost Matrix

	Actual negative	Actual positive
Predict negative	$C(0,0)$	$C(0,1)$
Predict positive	$C(1,0)$	$C(1,1)$

Table 2.2: Description of Experimental Data Sets

Data set	Size	# of attributes	# of categorical attributes	# of continuous attributes	missing data	Class distribution (class 0/class 1)	Class distribution for class 1 (%)
Chess	3196	36	36	0	N	1669/1527	47.78
Mushroom	8124	22	22	0	Y	4208/3916	48.2
tic-tac-toe	958	9	9	0	N	626/332	34.66
Ionosphere	351	34	0	34	N	225/126	35.90
magic gamma telescope	19020	10	0	10	N	12332/6688	35.16
image segmentation	2310	19	0	19	N	1980/330	14.29
Annealing	798	16	10	6	Y	710/88	11.03
car evaluation	1728	6	6	0	N	1210/518	29.98

Table 2.3: Cost Matrix Generated from Random Number Generator

cost matrix	$C(1,0)$	$C(0,1)$
1	1.011261	6.072268
2	2.739738	8.278665
3	4.152623	9.063662
4	5.784967	6.140660
5	6.415876	6.464492

Table 2.4: Comparison of Misclassification Costs Averaged by Five Cost Matrices

Data set	CNB	RUS	ROS	MetaCost	ICSNB	CNB/ ICSNB	RUS/ ICSNB	ROS/ ICSNB	MetaCost/ ICSNB
Chess	636.24	739.45	757.51	1178.61	533.03	16.22%	27.92%	29.63%	54.77%
Mushroom	71.47	87.36	67.35	1323.38	59.26	17.09%	32.17%	12.02%	95.52%
tic-tac-toe	501.06	584.56	580.30	532.22	467.45	6.71%	20.03%	19.45%	12.17%
Ionosphere	63.93	78.66	67.98	255.13	63.49	0.69%	19.28%	6.61%	75.11%
magic gamma telescope	8355.62	8140.43	8123.74	9602.78	8071.78	3.40%	0.84%	0.64%	15.94%
image segmentation	144.00	209.46	196.74	380.84	142.55	1.01%	31.94%	27.54%	62.57%
Annealing	127.23	225.54	185.60	183.72	124.27	2.33%	44.90%	33.04%	32.36%
car evaluation	186.74	328.28	290.23	183.72	164.13	12.11%	50.00%	43.45%	10.66%

Table 2.5: Average Misclassification Costs for Two-class Data Sets

	Cost Matrix	CNB	RUS	ROS	MetaCost	ICSNB	CNB/ ICSNB	RUS/ ICSNB	ROS/ ICSNB	MetaCost/ ICSNB
Chess	1	264.76	347.97	345.04	379.08	152.12	42.55%	56.29%	55.91%	59.87%
	2	572.34	786.22	791.70	1004.03	390.71	31.73%	50.31%	50.65%	61.09%
	3	742.68	994.93	1027.53	1438.16	571.98	22.98%	42.51%	44.33%	60.23%
	4	765.22	746.81	794.80	1470.94	751.06	1.85%	*-0.57%*	5.50%	48.94%
	5	836.17	821.33	828.48	1600.84	799.30	4.41%	2.68%	3.52%	50.07%
Mushroom	1	31.51	45.57	33.22	1088.99	20.13	36.11%	55.82%	39.40%	98.15%
	2	71.00	91.32	67.59	1507.82	47.58	32.99%	47.90%	29.61%	96.84%
	3	91.18	124.80	92.76	1675.82	66.47	27.10%	46.74%	28.34%	96.03%
	4	79.37	86.63	72.76	1144.33	78.77	0.75%	9.08%	*-8.26%*	93.12%
	5	84.31	88.48	70.43	1199.92	83.34	1.15%	5.81%	*-18.32%*	93.05%
Tic-tac-toe	1	165.30	212.72	213.02	213.02	130.21	21.23%	38.79%	38.87%	38.87%
	2	484.03	557.67	555.21	530.97	449.83	7.07%	19.34%	18.98%	15.28%
	3	663.06	762.90	749.42	713.23	637.90	3.79%	16.38%	14.88%	10.56%
	4	581.54	673.63	669.91	588.61	544.68	6.34%	19.14%	18.69%	7.46%
	5	611.35	715.89	713.93	615.25	574.62	6.01%	19.73%	19.51%	6.60%
Ionosphere	1	44.93	47.19	48.32	55.76	44.73	0.45%	5.20%	7.43%	19.78%
	2	67.54	87.63	70.83	82.09	66.99	0.81%	23.55%	5.42%	18.40%
	3	77.36	102.17	82.87	96.99	77.20	0.20%	24.44%	6.84%	20.40%
	4	62.76	76.74	66.32	83.02	62.11	1.04%	19.06%	6.35%	25.19%
	5	67.08	79.56	71.58	85.71	66.43	0.97%	16.50%	7.20%	22.50%
Magic gamma telescope	1	3728.38	4126.94	4134.92	3821.56	3829.08	*-2.70%*	7.22%	7.40%	*-0.20%*
	2	7712.31	8391.25	8413.80	7869.77	7788.22	*-0.98%*	7.19%	7.44%	1.04%
	3	10469.76	9856.08	9787.66	11471.42	10498.68	*-0.28%*	*-6.52%*	*-7.26%*	8.48%
	4	9634.57	8791.93	8749.46	12029.80	8831.15	8.34%	*-0.45%*	*-0.93%*	26.59%
	5	10233.09	9535.95	9532.85	12821.33	9411.74	8.03%	1.30%	1.27%	26.59%

Table 2.6: Average Misclassification Costs for Multiclass Data Sets

	Cost Matrix	CNB	RUS	ROS	MetaCost	ICSNB	CNB/ ICSNB	RUS/ ICSNB	ROS/ ICSNB	MetaCost/ ICSNB
Image segmentation	1	46.42	79.94	56.48	125.19	46.32	0.22%	42.06%	18.00%	63.00%
	2	112.47	162.60	148.09	291.37	112.06	0.37%	31.08%	24.33%	61.54%
	3	161.20	231.72	220.41	399.69	160.74	0.28%	30.63%	27.07%	59.78%
	4	191.39	272.76	266.47	516.31	188.32	1.60%	30.96%	29.33%	63.53%
	5	208.55	300.26	292.25	571.65	205.31	1.55%	31.62%	29.75%	64.08%
Annealing	1	47.94	104.51	65.23	74.38	47.13	1.69%	54.90%	27.74%	36.63%
	2	107.08	203.31	153.86	174.65	104.30	2.59%	48.70%	32.21%	40.28%
	3	144.72	266.49	215.28	228.44	137.78	4.79%	48.30%	36.00%	39.69%
	4	161.29	263.97	236.25	208.43	159.60	1.05%	39.54%	32.45%	23.43%
	5	175.14	289.43	257.36	232.69	172.54	1.48%	40.39%	32.96%	25.85%
Car evaluation	1	82.11	304.90	83.28	83.28	67.55	17.73%	77.84%	18.88%	18.88%
	2	190.69	225.62	225.62	225.62	169.73	10.99%	24.77%	24.77%	24.77%
	3	237.95	341.14	341.97	341.97	237.74	0.09%	30.31%	30.48%	30.48%
	4	197.08	378.05	395.11	290.08	167.63	14.94%	55.66%	57.57%	42.21%
	5	225.89	391.69	405.16	334.68	178.00	21.20%	54.56%	56.07%	46.81%

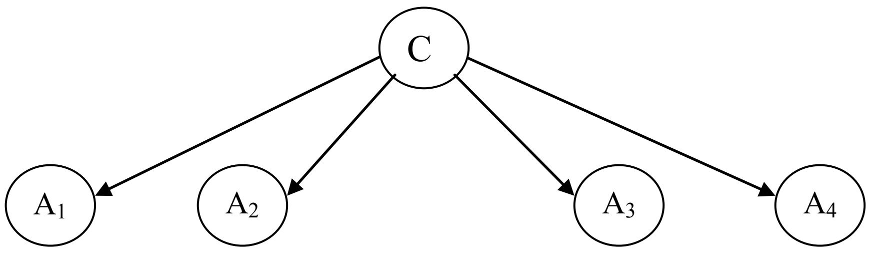


Figure 2.1: An Illustration of Conditional Independence

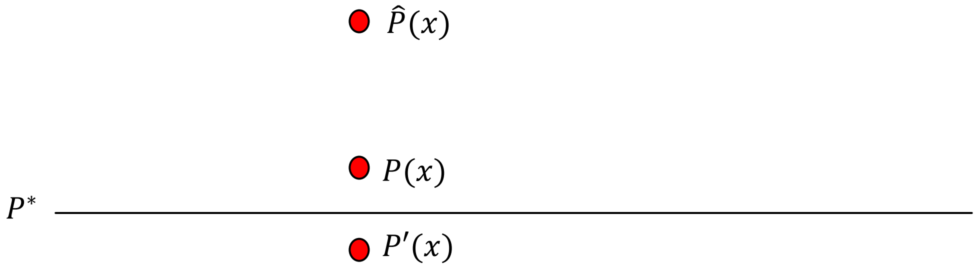


Figure 2.2: Poor Probability Estimation ≠ Poor Classification Decision

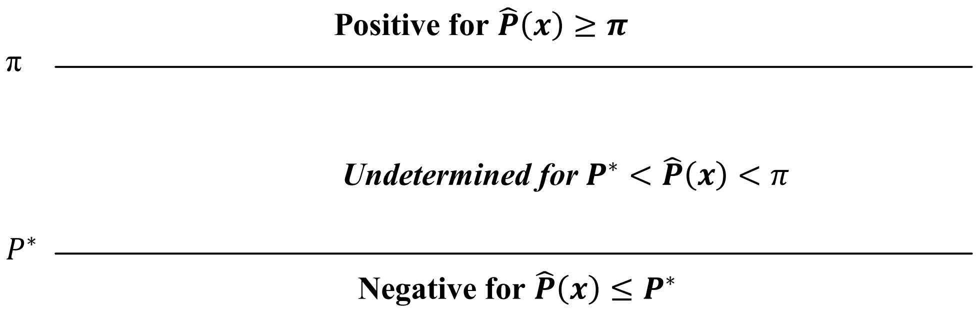

Positive for $\hat{P}(x) \geq \pi$
π
Undetermined for $P^* < \hat{P}(x) < \pi$
P^*
Negative for $\hat{P}(x) \leq P^*$

Figure 2.3: Label Instances When $\pi > P^*$

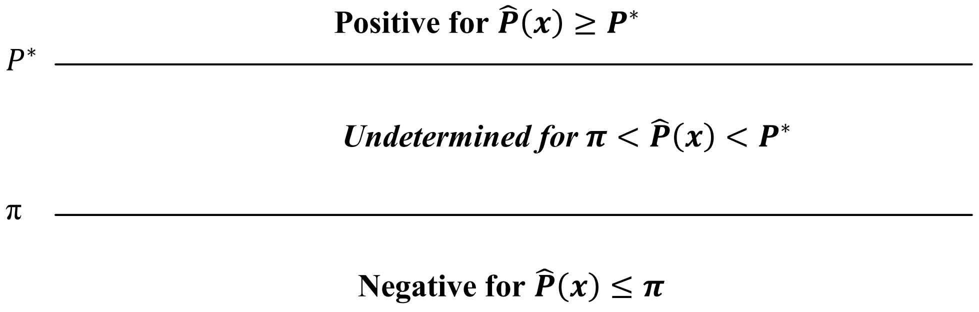

Positive for $\hat{P}(x) \geq P^*$
P^*
Undetermined for $\pi < \hat{P}(x) < P^*$
π
Negative for $\hat{P}(x) \leq \pi$

Figure 2.4: Label Instances When $\pi < P^*$

```
ICSNB($L,T$)
        $L$: training data set
        $T$: test data set

$U = \emptyset$.
NB($L,T$).        //Run the Naïve Bayes method
For each instance $x$ in $T$
        If ($\pi > P^*$ and $\hat{P}(x) \leq P^*$) or ($\pi < P^*$ and $\hat{P}(x) \leq \pi$)
                Label $x$ as negative.        //By Lemmas 1 and 2
        Else if ($\pi > P^*$ and $\hat{P}(x) \geq \pi$) or ($\pi < P^*$ and $\hat{P}(x) \geq P^*$)
                Label $x$ as positive.        //By Lemmas 1 and 2
        Else
                Add $x$ to $U$.    //Add unlabelled instances to $U$
        End if
End for
If ($U = \emptyset$)
        Terminate.
Else if ($|U| = |T|$)
        Hard($U$).    //Process hard instances
        Terminate.
Else
        $L$ = NN($U$, $k$).   //Construct the training data set for $U$
        $T$ = $U$.
        ICSNB($L,T$).
End if
```

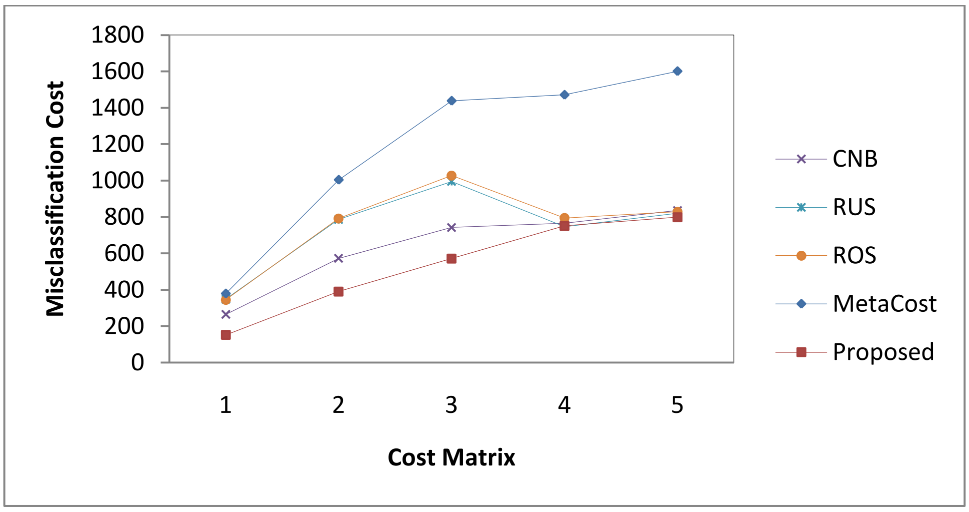
Figure 2.5: The Iterative Cost-sensitive Naïve Bayes Algorithm

```
Hard ($U$)
        $U$: the set of hard and unlabelled instances

For each instance $x$ in $U$
        If ( $\hat{P}(x) \geq P^*$)
                Label $x$ as positive.
        Else
                Label $x$ as negative.
        End if
End for
```

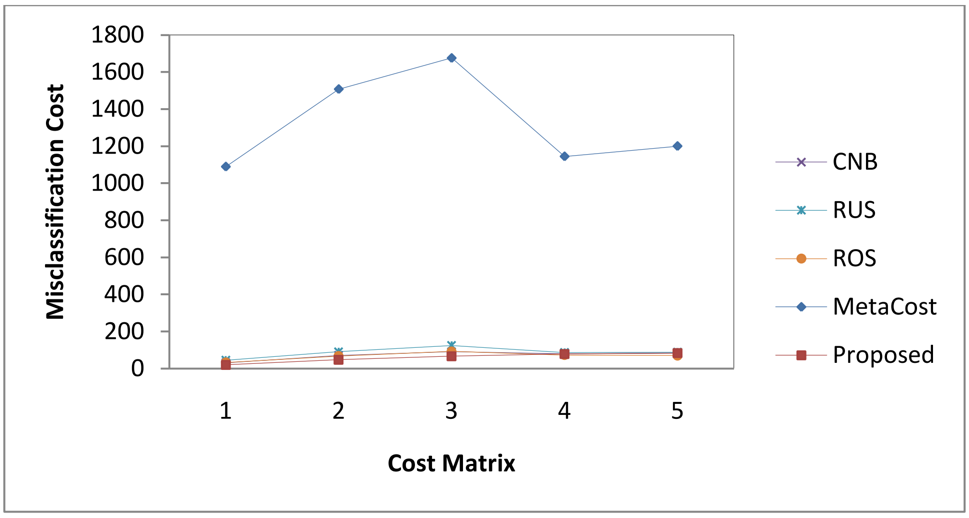
Figure 2.6: Procedure Hard()

```
NN ($U$, $k$)
        $U$: the set of unlabeled instances
        $k$: the number of nearest neighbors

$L_c = \emptyset$.    // $L_c$: the constructed training data set for $U$
For each instance $x$ in $U$
        Find its $k$ nearest instances in the original training data set.
        Add the $k$ nearest instances to $L_c$.
End for
Return $L_c$.
```

Figure 2.7: Procedure NN()

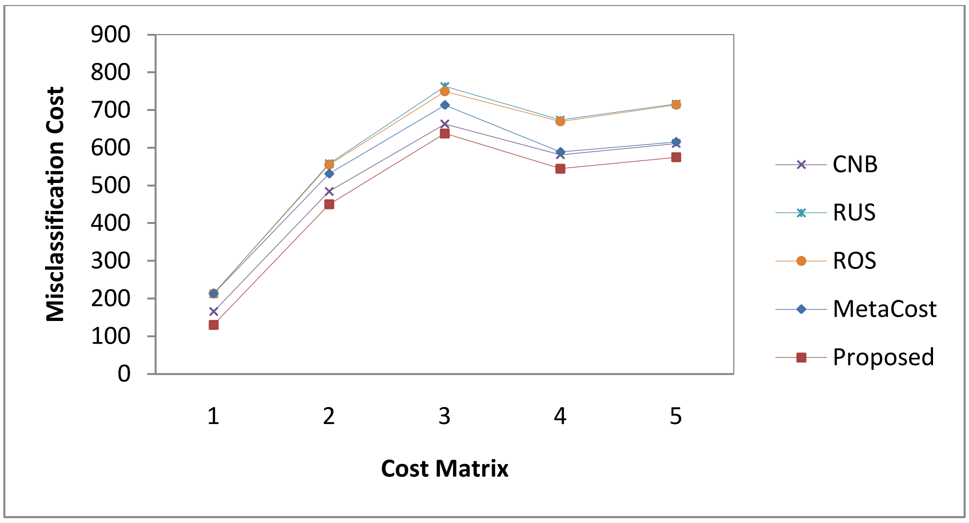


(a)

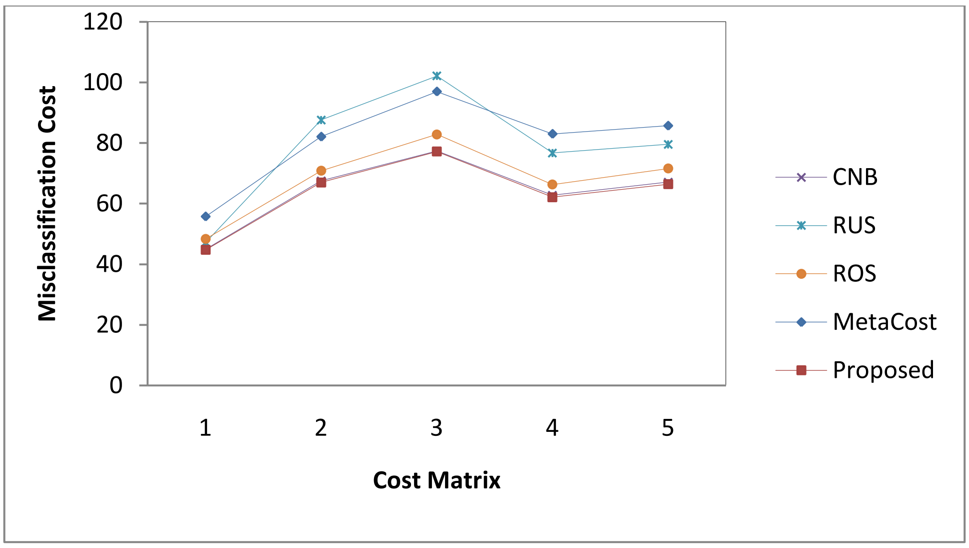


(b)

Figure 2.8: Cost Curve on Two-class Data Sets. (a) Chess; (b) Mushroom; (c) Tic-tac-toe; (d) Ionosphere; (e) Magic Gamma Telescope

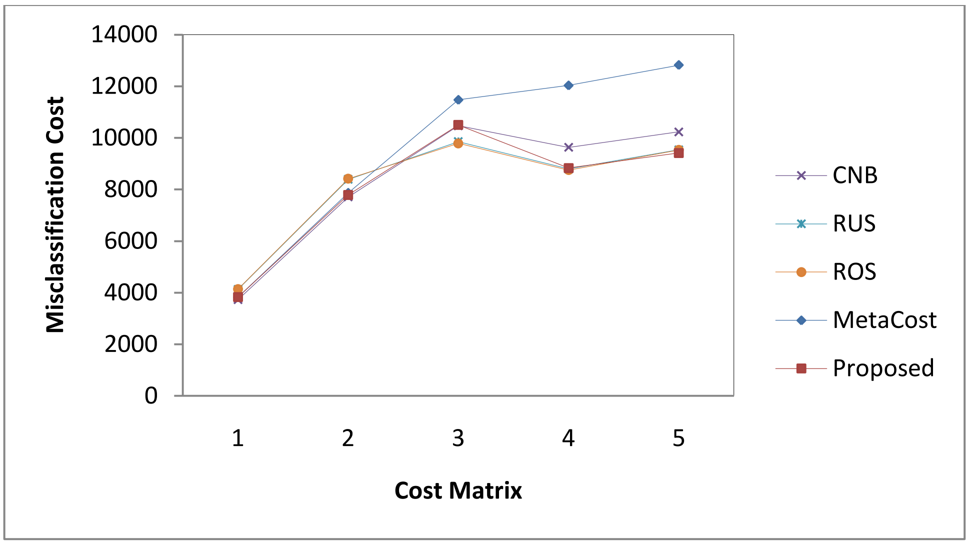


(c)



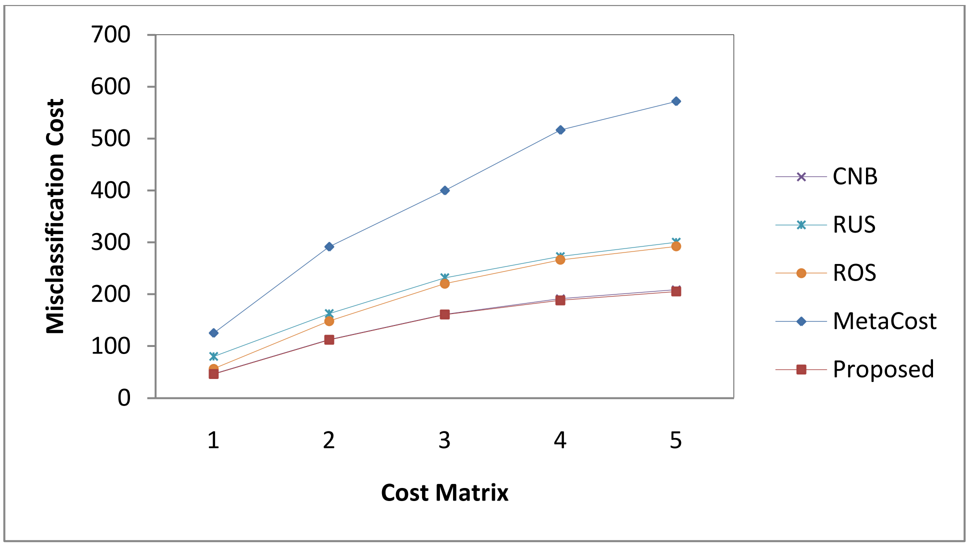
(d)

Figure 2.8: Continued



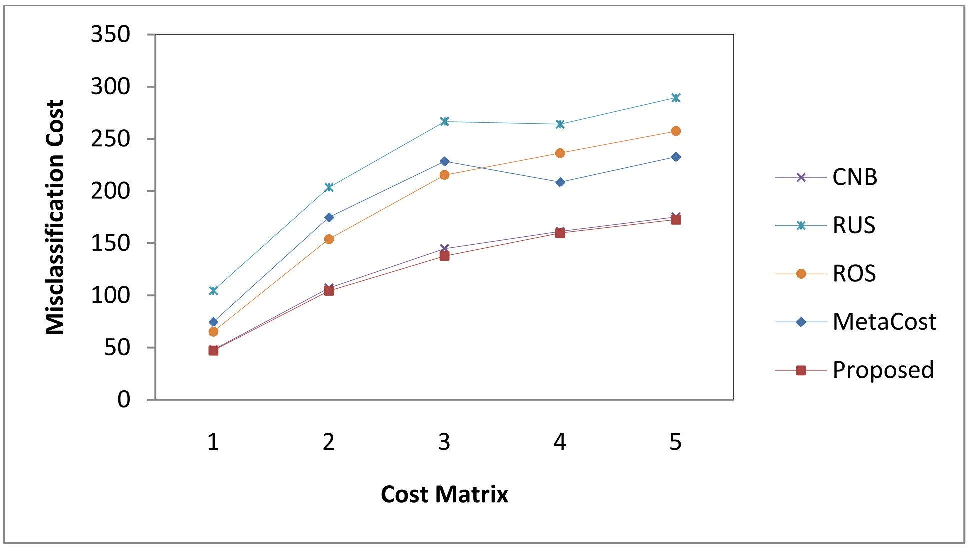
(e)

Figure 2.8: Continued

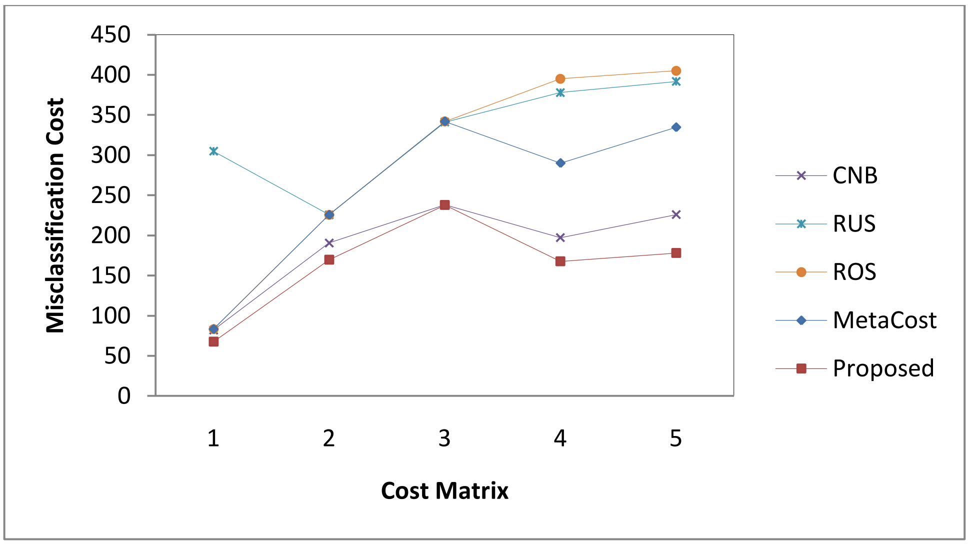


(a)

Figure 2.9: Cost Curve on Multiclass Data Sets. (a) Image Segmentation; (b) Annealing; (c) Car Evaluation



(b)



(c)

Figure 2.9: Continued

CHAPTER 3

COST-BASED UNDER-SAMPLING METHOD FOR IMBALANCE LEARNING

3.1 Introduction

In many real-world situations, making a managerial decision involves classifying an observation into one of several predefined groups. A typical case of this problem is binary classification in which an observation needs to be classified into one of two groups. This happens every day in our lives. For instance, a credit card company needs to make a decision whether an application should be approved. An Internet retailer wants to identify customers who will receive discount coupons as rewards based on their purchase histories. Besides business applications, other applications also face similar decision-making problems, such as disease diagnosis, detection of oil spills, detection of DDOS (Distributed Denial of Service), etc. The solution for solving this type of problem was first proposed by Fisher (1936) using a statistic method, Discriminant Analysis (DA). Since then, many methods have been developed for classification purposes. Today, various Discriminant Analysis methods still play significant roles in this area.

With the development of computer technology in both hardware and software, especially with the emergence of Internet technology, large amounts of data and information are kept in database systems and it has become much easier to access data.

Moreover, more accurate decisions can be made with the greater data availability and accessibility. Thus, a new technique, data mining, has been found to be very effective in solving these types of decision-making problems. People have discovered significant advantages to using the machine learning method in terms of effectiveness and efficiency, especially when handling a large amount of data. Derived from machine learning techniques, data mining is an integration of machine learning, computer visualization, and statistics. Among all the methods in data mining, classification learning is specialized to deal with the decision-making problem.

The underlying assumption of different classification algorithms is equal distributions and error costs among categories. However, this assumption is not consistent with the real business world. When we want to predict a purchase decision from an individual level for an existing customer, one inevitable problem is that customers with repeated purchase behavior are rare events for the whole customer base (King and Zeng 2000). Similarly, in the medical diagnosis domain, sick patients are always a small group when compared to the entire population. We regard this type of imbalance as a class imbalance problem, and the data set is called imbalanced data, where the size of one class overwhelms that of the other class (Barandela et al. 2004; Guo and Vikto 2004; Japkowicz and Stephen 2002). Such class imbalance violates the basic assumption for applying traditional classification methods. When a data set is imbalanced, traditional predictive models and methods, such as classification methods, tend to favor the majority class, resulting in high overall accuracy. But, in such cases, detection rates with respect to the minority class are often not satisfactory. For instance, when a model is trained on a binary data set with 1% of its examples from the minority class, a 99%

accuracy rate can be achieved by classifying all examples as belonging to the majority class. While 99% accuracy is often considered excellent, such a model often has no practical value since our interests are often on the minority class. In the real world, there are many examples where people are more interested in the minority class, not the majority class. For example, the majority of consumers simply ignore advertisement mails. But, the purpose of a mail campaign is to identify and focus on the minority of consumers who will respond to advertisement mails and make purchases subsequently, i.e., the minority class. The importance of the minority class and the existence of the imbalance problem in the real world encourage me to take data imbalance into consideration when building classification models.

Given any data set, the second inevitable problem in building any predictive model is the noise of data. Noisy data can have a negative impact on classification performance and degrade the generalizability of a prediction model. It usually shows its effect through affecting class prediction of its neighbor instances away from their true class labels. Noisy instances can also negatively influence deciding decision boundaries, thus putting other instances into wrong groups. Due to these reasons I just discussed, identifying noisy instances and removing them from input data sets will be crucial in building accurate classification models.

Furthermore, in many real-world applications, equal error cost cannot hold as well. Usually, misclassifying an instance belonging to the minority class incurs higher cost than misclassifying an instance belonging to the majority class. For example, misclassifying a potential purchaser as a nonpurchaser costs the entire revenue that could be realized from the potential purchaser. In comparison, the cost of misclassifying a nonpurchaser as a potential purchaser costs only the production and delivery of an advertisement mailing. In certain situations, the difference between misclassification costs can be even larger. For instance, diagnosing a cancer patient to be healthy can be a fatal mistake since a patient can lose his or her life because of the delay in this incorrect diagnosis and treatment.

Because of the violation of assumptions for the traditional classification method, it is necessary to seek a solution in building classification models sensitive to imbalance, noise, and cost issues. In this proposal, I develop a cost-based under-sampling method to significantly reduce the negative effects of those issues on the classification model.

3.2 Literature Review

3.2.1 Data Resampling

To solve data imbalance problems in predictive models using traditional classifiers, the first direction is to change the data distribution and then create a balanced data set. Changing data distribution can be done through two ways: under-sampling and over-sampling. The under-sampling method removes majority instances from a data set until the sizes of two groups are equal, while the over-sampling method generates more minority instances and keeps the majority group unchanged. These two methods are called data resampling. To do data resampling, usually we can have two approaches, random resampling and intelligent resampling. In random over-sampling (ROS), instances of the minority class are randomly duplicated, while instances of the majority class are randomly discarded from the data set for random under-sampling (RUS).

Some researchers have also proposed intelligent resampling methods. Kubat and Matwin (1997) proposed a technique called one-sided selection (OSS), which is considered to be the most effective method for under-sampling and is used as baseline method in this dissertation. One-sided selection attempts to intelligently under-sample the majority class by removing the majority class examples that are considered to be either redundant or noisy. Barandela et al. (2004) used KNN techniques to classify each instance in the training set using all the remaining data and removing those majority class instances that are misclassified. For the intelligent over-sampling method, Chawla et al. (2002) proposed the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE adds new, artificial minority instances by operating in feature space rather than data space. The synthetic samples are generated in the following way: first, SMOTE takes the difference between the feature vector (sample) under consideration and its nearest neighbor; second, this difference is multiplied by a random number between 0 and 1; third, the multiplied outcome is added to the feature vector under consideration. Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen. SMOTE has been demonstrated to perform well in several research studies (Zhou and Liu 2006; Hulse 2007).

However, there are limitations for both over-sampling and under-sampling methods. Random over-sampling may increase the likelihood of occurring overfitting, since it makes exact copies of the minority class examples. In this way, a symbolic classifier, for instance, might construct rules that are seemingly accurate, but actually cover one replicated example (Batista el al. 2004). Another limitation associated with over-sampling is increasing computation cost with larger data size. Considering the computing efficiency, people are always trying to select representative data examples and reduce sample size. But, over-sampling could increase sample size greatly for extremely skewed data sets. On the other hand, the major drawback of random under-sampling is that this method can discard potentially useful data that could be important for the induction process. Although researchers recommend intelligent resampling methods instead of random resampling to ease those limitations mentioned above (Batista et al. 2004), even the intelligent under-sampling, such as OOS, still cannot make sure it does identify all useless majority instances.

3.2.2 Cost-sensitive Learning

3.2.2.1 Introduction to cost-sensitive learning: Classification is the most important task in inductive learning and machine learning. A classifier can be trained from a set of training examples with class labels, and can be used to predict the class labels of new examples. The class label is usually discrete and finite. Many effective classification algorithms have been developed, such as Naïve Bayes, Decision Trees, Neural Networks, and so on. However, most raw classification algorithms pursue minimization of the error rate, i.e., lowering the percentage of the incorrect prediction of class labels. They ignore the difference among misclassification errors. In particular, they implicitly assume that all misclassification errors cost equally.

As I discussed already, in many real-world applications, this assumption is not true. The difference between different misclassification errors can be quite large. For example, in the medical diagnosis of a certain cancer, if the cancer is regarded as the positive class, and noncancer (healthy) as the negative, then, missing a cancer (the patient is actually a positive sample but is classified as a negative sample; thus, it is also called a “false negative”) is much more serious (thus, expensive) mistake than the false-positive error. The patient could lose his/her life because of the delayed treatment caused by the incorrect diagnosis.

There are a large variety of different types of costs in data mining and machine learning, including misclassification costs, data acquisition costs (instance costs, attribute costs, and labeling costs), computation costs, human-computer interaction costs, and so on (Turney 2000). Cost-sensitive learning considers a variety of costs in various components and processes of learning (Turney 2000), with the goal of minimizing the individual costs or the total cost. It is one of the most active and important research areas in data mining and machine learning and plays an important role in real-world data mining and machine learning applications.

The two most important types of costs are identified as misclassification costs and data acquisition costs. Misclassification costs are an extension of error rate, since different types of errors (such as false positives and false negatives) can have different costs. Data acquisition costs reflect how expensive it is to acquire extra information for assisting classification or building more accurate learning models. Recent works have considered methods of acquiring attribute values during training (Melville et al., 2004; 2005) and testing (Ling et al., 2004; Chai et al., 2004), for the purpose of reducing the misclassification cost. Melville et al. (2004) studied how to achieve the desired model accuracy by acquiring missing values in identified training examples with minimum cost. However, they did not minimize the total cost (misclassification and attribute costs) of testing examples with missing values. Actually, in many real-world applications, especially in the e-commerce area, it is often not difficult to acquire training samples and label samples. Hence, I only consider misclassification cost in this dissertation.

Elkan (2001) describes how the misclassification cost plays its essential role in various cost-sensitive learning algorithms. Without loss of generality, I assume binary classification (i.e., positive and negative class) in this chapter. In cost-sensitive learning, the costs of false positives (actual negative but predicted as positive; denoted as (*FP*)), false negatives (*FN*), true positives (*TP*) and true negatives (*TN*) can be given in a cost matrix, as shown in Table 3.1. In the table, I also use the notation $C(i, j)$ to represent the misclassification cost of classifying an instance from its actual class j into the predicted class i. (I use 1 for positive, and 0 for negative). These misclassification cost values can be given by domain experts, or learned via other approaches. In cost-sensitive learning, it is usually assumed that such a cost matrix is given and known. For multiple classes, the cost matrix can be easily extended by adding more rows and more columns.

Note that $C(i, i)$ (*TP* and *TN*) is usually regarded as the "benefit" (i.e., negated cost) when an instance is predicted correctly. In addition, cost-sensitive learning is often used to deal with data sets with very imbalanced class distribution (Japkowicz and Stephen, 2002). Usually (and without loss of generality), the minority or rare class is regarded as the positive class. It is often more expensive to misclassify an actual positive example into negative than an actual negative example into positive. That is, the value of *FN* or $C(0,1)$ is usually larger than that of *FP* or $C(1,0)$.

Given the cost matrix, an example should be classified into the class that has the minimum expected cost. This is the minimum expected cost principle. The expected cost $R(i|x)$ of classifying an instance x into class i (by a classifier) can be expressed as:

$$R(i|x) = \sum_j P(j|x)C(i,j) \qquad (3.1)$$

where $P(j|x)$ is the probability estimation of classifying an instance into class j. That is, the classifier will classify an instance x into positive class if and only if:

$$P(0|x)C(1,0) + P(1|x)C(1,1) \leq P(0|x)C(0,0) + P(1|x)C(0,1) \qquad (3.2)$$

Traditional cost-insensitive classifiers are designed to predict the class in terms of a default, fixed threshold of 0.5. Elkan (2001) shows that we can "rebalance" the raw training examples by sampling so that the classifiers with 0.5 thresholds are cost-sensitive. The rebalance is done as follows. If we keep all positive examples (as they are assumed as the rare class), then, the number of remained negative examples should be the number of raw negative examples multiplied by $C(1,0)/C(0,1) = FP/FN$, since usually $FP < FN$, the multiple is less than 1. This is thus often called "under-sampling the majority class." This is also equivalent to "proportional sampling," where positive and negative examples are sampled by the ratio.

3.2.2.2 Cost-sensitive learning methods: Broadly speaking, methods of cost-sensitive learning can be categorized into two categories. The first one is to design classifiers that are cost-sensitive in themselves. We call it the direct method. Examples of direct cost-sensitive learning are ICET (Turney 1995) and the cost-sensitive decision trees (Drummond and Holte 2000; Ling et al. 2004). The other method is to design a "wrapper" that converts any existing cost-insensitive classifiers into cost-sensitive ones.

The wrapper method is also called the cost-sensitive meta-learning method.

The main idea of building a direct cost-sensitive learning algorithm is to directly introduce and utilize misclassification costs into the learning algorithms. There are several studies on direct cost-sensitive learning algorithms, such as ICET (Turney 1995) and cost-sensitive decision trees (Ling et al. 2004). ICET (Turney 1995) incorporates misclassification costs in the fitness function of genetic algorithms. On the other hand, cost-sensitive decision trees (Ling et al. 2004) use the misclassification costs directly in their tree building process.

The cost-sensitive decision tree is a C4.5-like decision tree learning algorithm. It uses the minimal total cost (or maximum cost reduction) as a tree-split criterion, similar to maximum information gain ratio. Traditional Decision Tree C4.5 uses the information gain to select the best attribute to split training data. The information gain is defined as the difference between the entropy before splitting and that after splitting. The goal of the traditional Decision Tree C4.5 is to minimize the sum of the entropy of all leaves in the tree. Instead of minimizing entropy in attribute selection as in C4.5, the cost-sensitive decision tree selects the best attribute to split input data based on the expected total cost reduction. That is, an attribute is selected as a root of the (sub)tree if it minimizes the total cost, which is the sum of misclassification costs and attribute costs. The cost reduction is defined as the difference between the misclassification costs before splitting and the sum of the misclassification costs and the test cost of all the examples.

3.2.2.3 Threshold adjusting: Adjusting decision threshold to account for differential misclassification costs has been proposed and discussed by several researchers (Domingos, 1999; Provost 2000; Provost and Fawcett 2001), in which a new threshold instead of 0.5 is used to classify examples into positive or negative using. *MetaCost* (Domingos 1999) is a *threshold adjusting* method. It first uses bagging on decision trees to obtain reliable probability estimations of training examples, relabels the classes of training examples, and then uses the relabeled training instances to build a cost-insensitive classifier. Provost and Fawcett (2001) proposed a ROC convex hull method by combining ROC analysis with decision analysis for comparing the performance of a set of classifiers and identifying the optimal classifier or a subset of potentially optimal classifiers.

As I show in (3.2), we can have (3.3) if $p = P(j = 1|x)$

$$(1-p)c(1,0) + pc(1,1) \le (1-p)c(0,0) + pc(0,1) \qquad (3.3)$$

So, the threshold for making optimal decision is p* such that

$$(1-p^*)c(1,0) + p^*c(1,1) = (1-p^*)c(0,0) + p^*c(0,1) \qquad (3.4)$$

and then

$$p^* = \frac{c(1,0) - c(0,0)}{c(1,0) - c(0,0) + c(0,1) - c(1,1)} \qquad (3.5)$$

If we have equal misclassification cost, which means $c(1,0) = c(0,1)$, then $p^* = 0.5$. If p is greater than p*, the optimal prediction would be class 1. However, we usually have different misclassification costs and then p* is different from 0.5. Especially, in the real world, it is more likely that $c(1,0) < c(0,1)$ and $p^* < 0.5$. If we still use the default threshold of 0.5, the minority instances will be very difficult to be discovered. Therefore, we need to change the threshold of standard classifiers to meet the requirement of different cost ratio so that more minority instances can be identified under lower optimal threshold. This is how threshold adjusting works to meet the different misclassification costs.

3.2.3 Research Gap Analysis

I have discussed the three problems associated with data collected from the real world and used them to build predictive models. Among these three problems, imbalance is the main reason that causes low prediction performance of predictive models. To address the imbalance problem, researchers have proposed different resampling techniques to reduce the effect due to the under-represented interesting class. However, the limitation associated with the data resampling approach is that it might lose important or useful information in under-sampling and bring noise and redundant information into the data set in over-sampling. Especially for under-sampling, how to discover and remove redundant and noisy instances is the key issue that needs to be solved. However, even the most popular under-sampling method (OSS) is not good enough for removing noisy instances. The majority instances participating Tomek link may be useful when they are close to the decision boundary.

Taking unequal misclassification costs into consideration has also been shown to be a good solution to the imbalanced data problem. However, cost-sensitive learning does not change the training data. Thus, noisy or redundant data still exist. If we can incorporate the cost information into the process of identifying noisy instances, we may solve these limitations together. To my knowledge, there has been no research incorporating misclassification cost into the under-sampling method. This dissertation proposes a novel method in the under-sampling method: addressing the noisy and imbalanced data problems while being aware of different misclassification costs.

3.3 Cost-based Under-sampling Method

The noisy and the imbalanced data problems are closely related to each other. Noisy instances are more apt to be misclassified than others (Kubat and Matwin 1997; Kermanidis 2009). The negative effect of noisy instance can be demonstrated in Figure 3.1. The predicted class labels of the examples close to noisy examples can be different from their actual class labels because of the noisy examples. The objective of under-sampling is to remove the effect of noisy instances on classification learning. Removing all existing noisy instances from a given data set does not ensure a better classification performance for an imbalanced data set. On the one hand, even after removing noisy instances, the remaining data set might still suffer from the imbalance problem. On the other hand, the performance of a predictive model still suffers from noisy instances that remain in a balanced data set if we choose random under-sampling. Therefore, it is desirable to handle these two problems simultaneously. Specifically, I will determine the noise level of each majority instance in a data set first and then balance the data set by removing majority instances with top noise levels.

The noise level of an instance refers to how ineffective the instance classifying of other instances in a data set is. Particularly, the noise level of an instance A is determined by (a) the number of instances misclassified by A and (b) the number of instances correctly classified by A. I determine the noise level of an instance based on prior research on the nearest neighbor classification method (Cover and Hart 1967). For an instance A, let A.class denote the class A belonging to and A.class $\in$ {majority, minority}. If the nearest neighbor of an instance B is A, by the nearest neighbor classification method, the predicted class of B is A.class. If A.class $\neq$ B.class, instance B is misclassified by instance A. Otherwise, instance B is correctly classified by instance A. Given that an instance A misclassifies n instances and correctly classifies m instances, where n and m are nonnegative integers, the noise level of A is measured as n-m.

The noise level of an instance is further refined by incorporating different misclassification costs. Let c[majority][minority] denote the cost of misclassifying a majority instance as minority and c[minority][majority] denote the cost of misclassifying a minority instance as majority. If A.class=majority, the noise level of A, A.noise, is measured as,

$$\text{A.noise} = n\times c[\text{minority}][\text{majority}] - m\times c[\text{majority}][\text{minority}] \qquad (3.6)$$

where nonnegative integers n and m denote the number of instances misclassified and correctly classified by A, respectively. In (3.6), n$\times$ c[minority][majority] represents the misclassification cost caused by A while m$\times$ c[majority][minority] represents the misclassification cost avoided due to A.

If any instance in a data set could better classify other instances in the data set (i.e., more instances are classified correctly), a classifier learned from the dataset would be able to classify new instances better. Therefore, the proposed under-sampling method outlined below calculates the noise level of each majority instance according to (3.6) and removes majority instances with top noise levels.

Below is the example showing the procedure of my proposed method. There are four examples in Figure 3.1, a, b, c, and A. A is the only nearest neighbor to a, b, and c in the data set. Data example c has the same class label as that of A while a and b have different labels.

Given the cost matrix in Table 3.2, I can calculate the noise level of A as 12*2-1=23. Similarly, I can calculate noise level for each majority in the given data set. Then, I can sort the minority examples based on their noise levels in descending order. To do under-sampling, I need to decide how many minority instances are to be removed and then drop minority instances based on their noise levels from high to low.

3.4 Empirical Experiment

3.4.1 Data Sets

In my empirical experiment, three data sets are used to evaluate my proposed method. The first two data sets were collected at a leading online retailer in the U.S. One is an online transaction data set, including the transaction information for about 100K customers in 2004 and repurchase activity in 2005 as well. The other is online search keyword data collected from the same company, which includes characteristics of 2,000 search keywords and revenue indicators for each keyword. The third data set is derived from the study of Pant and Sheng (2009). There are 2,694 competitive companies extracted from the Russell 3000 Index. For each of the companies, a fixed number of competitors and noncompetitors were identified using Hoover's API tools. There are 32,970 pairs of companies across 2,694 companies, half of which are competitors and the other half of which are noncompetitors.

The online transaction data set contains 99,998 individual customers and their purchase histories in twelve consecutive months from 1/1/2004 to 12/31/2004. To facilitate a marketing campaign launched by the online retailer, I used customers' purchase histories data to predict their repurchase behaviors. To this end, I took those purchase data in 2004 to generate predictors and used customers' repurchase activities in 2005 as class labels (i.e., repurchase in 2005 or not). Specifically, each record has twelve predictor attributes and one class label attribute. Predictor attributes reflect purchase information made in 2004, such as total purchase amount, return amount, recency of purchase, coupon used or not, etc. The repurchase rate is 26.4% for my data set. In other words, 26.4% of customers who purchased in 2004 repurchased in 2005. Therefore, the repurchase group is the minority class and the nonrepurchase group is the majority class. Table 3.3 shows the descriptive statistics of all twelve independent attributes for online transaction data.

For the online search keyword data, a web analytics tool, Omniture, is used to track users' activities and record information about how users reached the website from organic and paid search results as well as via what search queries they visit the web site (which consist of one or multiple keywords). The total revenues generated from a given search query are also reported, and the data are aggregated to weekly summaries. I randomly selected 2,000 search keywords (phrases) from those that appeared in sponsored search referrals during the one month prior to the start of the 10-week data collection period. For each search keyword (phrase), I identified whether this keyword (phrase) contained specific product information, store information, or product category information etc. The class label is whether these keywords can keep on generating revenue during the data collection period. Overall, only 17% keywords still generate revenue. I include the description and statistics of these attributes in Table 3.4 and Table 3.5, respectively.

The competitor identification data are obtained from the study by Pant and Sheng (2009). In this data set, each company is measure by five web metrics as described in Table 3.6. The related descriptive statistics is shown in Table 3.7. To train the predictive model, 66% of randomly selected data (pairs) are used as training data sets and the remaining data for testing. To maintain the disjointed nature of the training and testing data sets, the pairs of competitors from the testing data whose reverse instances appear in the training data were removed. The above process was repeated 50 times to generate 50-pair training and testing data sets. For each training or testing data set, 10% of company pairs indicate competitor relationships.

3.4.2 Experiment Results and Discussion

In the experiment, I compared my proposed method with OSS, a classic and effective under-sampling method, using six of the most popular classification methods: Bayesian Network, Decision Tree (C4.5), Logistics, neural network, and SVM. Weka was used to run predictive analysis. All experimental results were computed through 10-

fold cross validation (except for competitor data): one tenth of the experimental data set as test data and the rest as training data. Training data were under-sampled and balanced using the proposed method or OSS. A model was then learned on the under-sampled training data using different classifiers. The learned model was subsequently applied to test data and misclassification cost was finally calculated. The final performance is the average across the ten performance results.

Case study I - Target Marketing: Target Marketing is critical to the success of online retailers and has attracted researchers' interests for several decades. Prior research has proposed statistical models (Rossi et al. 1996; Sismeiro and Bucklin 2004) and more recently data mining-based methods (Kim et al. 2005; West et al. 1997) to predict consumer behaviors. For the purpose of predicting future behavior, historical data are usually used to train the predictive model. I first ran the analysis using the actual cost information obtained from the retailer. For the online retailer studied in this research, the average marketing cost spent on a customer is \$0.27 and the average revenue contributed by a customer is \$104.20. Hence, I set the cost of misclassifying a nonrepurchase customer at \$0.27 and the cost of misclassifying a repurchase customer at \$104.20. As shown in Table 3.8, my proposed method reduces misclassification cost by 30.08% on average when compared to OSS. I also report misclassification costs resulting when no under-sampling method is applied to training data in the "raw" column of Table 3.8. My proposed method reduces misclassification cost by 41.79% on average when compared to the "raw." This result verifies the improvement of under-sampling methods in reducing misclassification cost when dealing with noisy and imbalanced data.

I further examined whether my proposed method consistently outperforms OSS and raw data under different cost ratios. More experiments were conducted using different cost ratios (i.e., c[majority][minority]:c[minority][majority]). I repeated my experiments three times using artificial cost ratios 1:1, 1:10, and 1:100. The results are shown in Table 3.9 through Table 3.11.

From the above results, I find that the under-sample method, especially my proposed method, always has better performance under cost ratio greater than 1:1. This finding is consistent with the cost-sensitive learning method, which is more effective given a higher cost ratio, or more sensitive to a higher cost ratio.

In addition, more experiments were conducted to verify whether those instances removed by my method are high-cost instances. For this purpose, I combined the removed majority instances with the minority instances in the raw training set to form a new training set. Then I used these new training data sets to run classifier models and tested them on the same testing data sets. Figure 3.4 through Figure 3.7 present the results using the removed training data under cost ratio 1:1, 1:10, 1:100, and real cost.

I observed a similar pattern from the above experiment result. For low-cost ratio, the proposed method did not show any advantage in reducing misclassification cost compared to raw data and removed data. Actually, the proposed method even has the highest misclassification cost when compared to raw data sets and removed data sets. However, once the cost ratio is over 1:1, the proposed method outperforms the other two data sets persistently, which further verifies my previous conclusion that the proposed method is more effective in dealing with imbalance learning with high-cost ratio. In addition, the removed data set also shows better performance than the raw data set, which means the combined negative effect (raw data) is higher than the individual negative effect (removed data).

Although very informative, the above results cannot explain what the possible reason that can explain the performance of the proposed method is. Therefore, I compared the mean difference of each attribute across these three data sets (raw data set, remained data set, and removed data set) to see whether the performance comes from the change of the distribution of the data. From Table 3.12, I found that the attributes mean values of majority class in the remained data set is far from the ones of the minority class comparing to raw data sets and removed data sets. It indicates that my method has removed those majority instances surrounding minority instances. Through my method, the data set has come to have a clear boundary to differentiate minority instances from majority instances.

In the next two sections, I discuss the results of the proposed method based on the other two data sets, search engine marketing (SEM) and competitor identification, and three cost ratios, 1:1, 1:10 to 1:100, were used in the experiments.

Case study II - Search Engine Marketing (SEM): Search Engine Marketing (SEM) has become an interesting research topic in the last decade. Through search engines, Internet retailers can easily reach potential customers with less cost. So, effective SEM strategy becomes very critical to the success of Internet retailers and it helps them achieve higher click-through rates and even higher conversion rates. In this second study, a search keyword data collected from an Internet retailer is used to test my method. Similar to the first study, I first compared the performance of my method, OSS, and raw data measured by overall misclassification. Also, six classifiers have been used. The results are in Table 3.13 through Table 3.15.

Furthermore, I combined the removed majority instances with the minority instances in the raw training set to form a new training set. Figures 3.8 to 3.10 show the misclassification costs for three data sets under different cost ratios.

From all the above results, I can observe consistent result patterns with those I found in the online transaction data. The data set generated by the proposed method does not show any benefit as compared with the other two data sets under equal cost ratio. However, once the cost ratio is unequal, the proposed under-sampling method incurs the lowest misclassification cost. This result further verifies that my method is effective in lower misclassification cost when misclassification costs for two groups are different. Once again, Table 3.16 tells us that those majority instances close to minority are more likely to be identified and removed.

Case study III - Competitor Identification: In the third study, I followed the same procedure as I did for the previous two studies. The only difference is that the class distribution and competitor identification data have a higher distribution skewness of 10%. The misclassification cost comparison is shown in Tables 3.17 through 3.19 for three cost ratios, respectively. The observed patterns from previous studies still exist in this even more skewed (10%) data set.

Again, I used three training sets (raw, removed, proposed) to learn the test data and show the results in Figures 3.11 through 3.13. Similar patterns are observed to those in the previous two cases. The performance of my proposed method works better when the cost ratio is unequal.

Table 3.20 contains the comparison for mean of each attribute between raw training sets and remained training sets. The difference between majority class and minority class in the remained set is increased.

3.5 Additional Experiments

3.5.1 Experiment I

In the previous chapter, I tested my proposed method on three real-world data sets. The results show that the proposed method outperforms the others. Using my proposed method, I identified and removed those noisy majority instances from the training data set to minimize the overall learning cost in the testing dataset. I verified that the removed samples can incur high costs, as shown in Figures 3.4 through 3.7.

In this section, I conducted a new experiment to test the performance of my method on noisy data. For this purpose, the removed majority instances were combined with the minority instances from the training data set to form a new training data set. Using this newly formed training data set, I tested different data resampling methods on the same testing data set. This experiment is conducted based on online shopping data. Two under-sampling methods are compared in this experiment, proposed and RUS (random under-sampling) using the same classifiers. The comparison result is summarized in Table 3.21.

Based on Table 3.21, the proposed method is slightly better than RUS under cost ratio 1:1, but it has a much more significant improvement when cost ratio increases. Furthermore, I found that using the removed instances as new training data can incur higher overall misclassification costs for the same testing data set under unequal cost ratios.

3.5.2 Experiment II

In the previous experiments, I used three real data sets as a test bed and did not change class distribution. The true distributions of these three data sets are 10%, 17%, and 26.4%, respectively. Although the distributions have some diversity and the results show the excellent performance of the proposed method, it is not clear whether my approach can work well under diverse distributions or even skewed distribution. To further explore the performance of my proposed method in different distribution, I conducted another experiment using the online shopping data. Based on this data set, I controlled the number of majority examples in the training data set so as to change the class distribution. I manually removed the minority instances from training data sets and kept majority examples unchanged. The tested class distributions are 5%, 10%, 15%, and 20%, and only cost ratio 1:10 and 1:100 are used in the experiment. The results are summarized in Table 3.22 and Table 3.23.

Tables 3.22 and 3.23 show the misclassification costs of testing data sets using proposed method and random under-sampling for six different classifiers under different class distributions and cost ratios. Overall, my method generates lower cost when compared to RUS across all six classifiers. The result provides more support that the proposed method can achieve greater imbalance learning even with different class distributions.

3.5.3 Experiment III

As I discussed in Section 3.2, data resampling is an effective method in alleviating the negative effect of imbalanced class distribution. In data resampling, there are two approaches, under-sampling and over-sampling. For under-sampling, people remove instances from the majority group to create a balanced input while adding instances to the minority group in over-sampling. Which sampling method is better is still under research. Therefore, in this experiment, I compared the performance of my method with one popular over-sampling method, SMOTE (Chawla et al. 2002). In SMOTE, noisy or costly examples are not identified and synthetic minority instances will be generated between any two randomly selected minority instances. Since this method is not cost-sensitive, it should be worse if I use misclassification cost as measurement. The data set used in this experiment is still online shopping data and only here did I consider the natural class distribution. Six classifiers used before were also tested; the result is summarized in Table 3.24.

From Table 3.24, I can see that my proposed method outperforms the SMOTE under high-cost ratios, both 1:10 and 1:100. Given the equal-cost ratio, SMOTE is better than my method except for SVM. As I discussed above, SMOTE is not a cost-sensitive resampling method and the experiment result further verifies my judgment.

3.6 Conclusions and Limitations

In this study, I developed a cost-oriented under-sampling method to address negative effects of noisy instances and imbalanced data on predictive performance. I incorporated misclassification cost into the procedure of identifying noisy instance when conducting under-sampling. The proposed method was applied to three different online applications: target marketing, SEM, and competitor identification. Three real world data sets from these three applications were collected for the expirical tests. Extensive experiments including six classifiers have been done using my proposed approach. I also compared my results with ones using raw data and OSS, a widely used under-sampling approach. The results show that my approach outperforms others and demonstrates the effectiveness of my method in reducing overall misclassification cost incurred by noise and imbalance problems. The advantage of my approach is even greater when the cost ratio is higher.

In addition, I performed three more experiments to test if the proposed method has consistent performance in different situations. I first used the removed examples to form new training data and tested them on the same testing data. I found that the training data using the examples removed by proposed method always generates higher cost when compared to the one by random under-sampling. It indicates that my method can really identify and remove most costly examples. Second, I created synthetic data sets with different class distribution as training data sets. Again, the proposed method shows better performance when class distribution is changed. Last, I compared my method with the over-sampling method, SMOTE. The result still favors the proposed method in terms of misclassification cost.

The criteria used for removing majority instance were based on the noise level of each majority instance and the noise level is measured by how much misclassification cost it incurs. If the threshold of noise level is set to be zero, I may not get a balanced data set after removing all possible majority instances with noise levels greater than zero.

Either the majority class is still the majority class, or it becomes the minority class. To generate a balanced data set, in the first case, I may have to discard useful instances using random under-sampling. In the second case, I still have room to improve the performance. Therefore, first, an effective over-sampling method would be necessary and useful. It can be integrated with the current method to solve the problem I mentioned above. Especially for the high-skewed data set, there may not be enough data for the training model if majority instances are removed to make it balanced. Although there are some existing over-sampling techniques such as SMOTE, incorporating cost information into over-sampling deserves more attention.

Second, I have used three artificial cost ratios in the experiment. Even when I observed the change of performance when moving from equal cost ratio to differentiated cost ratios, it is not clear how this change happens across different cost ratios. Further efforts should pursue the effect of my method under more different cost ratios. Furthermore, in terms of the experiment, I also need to do comparison analysis between the proposed method and threshold adjusting or cost-sensitive learning methods.

Last but not least, these are also many multiclass applications in the real world. The "balance" in multiclass situation is more complicated than the two-class problem. It would be interesting to extend my method for solving the two-class data to the multiclass problem.

Table 3.1: An Example of Cost Matrix for Binary Classification

	Actual negative	Actual positive
Predicted negative	C(0,0), or TN	C(0,1), or FN
Predicted positive	C(1,0), or FP	C(1,1), or TP

Table 3.2: An Example of Cost Matrix

	Actual negative	Actual positive
Predict negative	0	12
Predict positive	1	0

Table 3.3: Descriptive Statistics of Online Transaction Data Set

Attribute	Mean	Std. Dev.	Min	Max
Duration_first_visit	280.70	325.83	0	2615
Duration_first_purchase	248.28	275.73	0	2075
Duration_registration	280.70	325.83	0	2615
Return_freq	0.09	0.33	0	12
Return_item_qty	0.15	9.71	0	3039
Return_fee	0.32	1.90	0	158.4
Refund_amt	9.18	112.56	0	27702.95
Total_purchase	170.41	737.67	0	170713.9
Purchase_freq	1.73	2.28	1	310
Recency_purchase	119.71	106.88	0	365
Num_cmpgn	1.70	2.23	0	310
Num_coupon	0.14	0.56	0	52

Table 3.4: Description of Independent Attributes for Online Search Keyword Data Set

Attribute	Description
word_count	Number of words for the keyword
Ret_spe	Retailer specific, 0 or 1
bnd_spe	Brand specific, 0 or 1
prd_spe	Product specific, 0 or 1
subc_spe	Subcategory of product, 0 or 1
mod_spe	Model or series of a product, 0 or 1
fea_spe	Feature related, 0 or 1
shp_spe	Shopping intention (including promotion related), 0 or 1

Table 3.5: Descriptive Statistics of Online Search Keyword Data Set

Attribute	Mean	Std. Dev.	Min	Max
word_count	3.30	1.52	1	14
ret_spe	0.07	0.26	0	1
bnd_spe	0.26	0.44	0	1
prd_spe	0.84	0.37	0	1
subc_spe	0.38	0.49	0	1
mod_spe	0.09	0.29	0	1
fea_spe	0.52	0.50	0	1
shp_spe	0.15	0.36	0	1

Table 3.6: Description of Web Metrics for Competitor Data Set

Web metrics	Description
sim_in	similarity in the web links that are directed towards web sites of companies of interest
sim_out	similarity in the web links that are going out from the sites of companies of interest
sim_text	Similarity of self-description of the sites of companies of interst
count_news	Number of news containing two companies's names
count_search	Number of search results containing two companies's names

Table 3.7: Descriptive Statistics of Web Metrics for Competitor Data Set

Web metrics	Mean	Std. Dev.	Min	Max
sim_in	0.01	0.04	0	0.97
sim_out	0.01	0.07	0	1.00
sim_text	0.03	0.05	0	0.89
count_news	5.55	88.05	0	4575
count_search	18776.42	163269.50	0	10500000

Table 3.8: Comparison of Misclassification Costs for
Re-balanced Online Transaction Data Set (real cost ratio)

Classifier	Misclassification Cost ($)			Relative % Improvement	
	Proposed	OSS	Raw	Proposed vs. OSS	Proposed vs. Raw
Bayesian Network	108859.05	140126.97	158972.84	-22.31%	-31.52%
Decision Tree	94393.79	142179.19	209876.29	-33.61%	-55.02%
Logistic	121096.32	189241.5	221477.92	-36.01%	-45.32%
Naïve Bayes	188461.30	205047.49	201419.5	-8.09%	-6.43%
Neural Network	98738.35	155185.67	216962.83	-36.37%	-54.49%
SVM	98738.35	124045.07	269537.95	-20.40%	-63.37%

Table 3.9: Comparison of Misclassification Costs for
Re-balanced Online Transaction Data Set (cost ratio 1:1)

Classifier	Misclassification Cost ($)			Relative % Improvement	
	Proposed	OSS	Raw	Proposed vs. OSS	Proposed vs. Raw
Bayesian Network	3317.80	2841.600	2661.600	16.76%	24.65%
Decision Tree	3480.30	2820.900	2386.700	23.38%	45.82%
Logistic	3037.90	3216.200	2398.000	-5.54%	26.68%
Naïve Bayes	2717.90	2555.300	2590.700	6.36%	4.91%
Neural Network	3141.00	2761.800	2381.200	13.73%	31.91%
SVM	3069.00	3173.500	2600.800	-3.29%	18.00%

Table 3.10: Comparison of Misclassification Costs for
Re-balanced Online Transaction Data Set (cost ratio 1:10)

Classifier	Misclassification Cost ($)			Relative % Improvement	
	Proposed	OSS	Raw	Proposed vs. OSS	Proposed vs. Raw
Bayesian Network	12656.00	14909.7	16365.9	-15.12%	-22.67%
Decision Tree	11710.50	15067.2	20505.5	-22.28%	-42.89%
Logistic	13426.10	19528.7	21521.2	-31.25%	-37.61%
Naïve Bayes	18966.00	20252	19972.4	-6.35%	-5.04%
Neural Network	16038.90	16135.8	21113.8	-0.60%	-24.04%
SVM	12362.20	13841.2	25881.1	-10.69%	-52.23%

Table 3.11: Comparison of Misclassification Costs for
Re-balanced Online Transaction Data Set (cost ratio 1:100)

Classifier	Misclassification Cost ($)			Relative % Improvement	
	Proposed	OSS	Raw	Proposed vs. OSS	Proposed vs. Raw
Bayesian Network	105887.20	135590.7	153408.9	-21.91%	-30.98%
Decision Tree	90327.50	137530.2	201693.5	-34.32%	-55.22%
Logistic	117314.00	182653.7	212753.2	-35.77%	-44.86%
Naïve Bayes	181507.90	197219	193789.4	-7.97%	-6.34%
Neural Network	98843.30	149875.8	208439.8	-34.05%	-52.58%
SVM	100427.00	120518.2	258684.1	-16.67%	-61.18%

Table 3.12: Comparison of Mean Difference for Online Transaction Data

Variable	Raw			Remained (real cost)			Removed	Remained (Majority) - Minority	Removed - Minority	Raw (Majority) - Minority
	Overall	Majority	Minority	Overall	Majority	Minority				
first_visit	280.696	257.852	343.763	297.565	251.084	343.763	261.6595	-26.96%	-23.88%	-24.99%
first_purchase	248.284	227.325	306.151	263.959	221.510	306.151	230.5961	-27.65%	-24.68%	-25.75%
registration	280.696	257.852	343.763	297.565	251.084	343.763	261.6595	-26.96%	-23.88%	-24.99%
rtntn_freq	0.087	0.074	0.124	0.094	0.064	0.124	0.079011	-48.39%	-36.28%	-40.32%
rtn_item_qty	0.148	0.138	0.175	0.126	0.077	0.175	0.171644	-56.00%	-1.92%	-21.14%
rtn_fee	0.324	0.270	0.474	0.345	0.215	0.474	0.300492	-54.64%	-36.61%	-43.04%
rfnd_amt	9.18	7.973	12.512	9.854	7.179	12.512	8.420221	-42.62%	-32.70%	-36.28%
total_purchase	170.406	140.109	254.053	190.933	127.428	254.053	147.2428	-49.84%	-42.04%	-44.85%
purch_freq	1.735	1.434	2.564	1.931	1.294	2.564	1.513207	-49.53%	-40.98%	-44.07%
recency	119.709	130.952	88.666	119.173	149.866	88.666	120.3131	69.02%	35.69%	47.69%
num_cmpgn	1.703	1.408	2.517	1.897	1.274	2.517	1.483727	-49.38%	-41.05%	-44.06%
num_coupon	0.141	0.098	0.257	0.164	0.070	0.257	0.113897	-72.76%	-55.68%	-61.87%

Table 3.13: Comparison of Misclassification Costs for Search Keyword Data (cost ratio 1:1)

Classifier	Misclassification Cost ($)			Cost Reduction by the Proposed Method	
	Proposed	OSS	Raw	Compared to OSS	Compared to Raw
Bayesian Network	71.60	63.40	41.80	12.93%	71.29%
Decision Tree	78.20	68.80	33.30	13.66%	134.83%
Logistic	74.20	59.70	33.00	24.29%	124.85%
Naïve Bayes	84.70	64.70	42.00	30.91%	101.67%
Neural Network	82.60	79.90	33.20	3.38%	148.80%
SVM	83.60	75.10	34.00	11.32%	145.88%

Table 3.14: Comparison of Misclassification Costs for Search Keyword Data (cost ratio 1:10)

Classifier	Misclassification Cost ($)			Cost Reduction by the Proposed Method	
	Proposed	OSS	Raw	Compared to OSS	Compared to Raw
Bayesian Network	137.60	178.60	223.60	-22.96%	-38.46%
Decision Tree	139.80	160.60	297.90	-12.95%	-53.07%
Logistic	139.10	160.50	302.10	-13.33%	-53.96%
Naïve Bayes	127.70	150.20	201.30	-14.98%	-36.56%
Neural Network	138.60	279.70	269.90	-50.45%	-48.65%
SVM	140.10	175.00	340.00	-19.94%	-58.79%

Table 3.15: Comparison of Misclassification Costs for Search Keyword Data (cost ratio 1:100)

Classifier	Misclassification Cost ($)			Cost Reduction by the Proposed Method	
	Proposed	OSS	Raw	Compared to OSS	Compared to Raw
Bayesian Network	731.20	1330.60	2041.60	-45.05%	-64.18%
Decision Tree	741.50	1078.60	2943.90	-31.25%	-74.81%
Logistic	696.30	1168.50	2993.10	-40.41%	-76.74%
Naïve Bayes	523.40	1005.20	1794.30	-47.93%	-70.83%
Neural Network	555.60	2277.70	2636.90	-75.61%	-78.93%
SVM	689.30	1174.00	3400.00	-41.29%	-79.73%

Table 3.16: Comparison of Mean Difference for Search Keyword Data

Variable	Raw			Remained (1:100)			Removed	Remained (Majority) - Minority	Removed - Minority	Raw (Majority) - Minority
	Overall	Majority	Minority	Overall	Majority	Minority				
length	21.030	22.195	15.341	20.624	26.033	15.341	21.236	69.70%	38.43%	44.68%
word_count	0.041	3.507	2.315	3.214	4.136	2.315	3.350	78.66%	44.71%	51.49%
ret_spe	-0.866	0.058	0.129	0.101	0.072	0.129	0.054	-44.19%	-58.14%	-55.04%
bnd_spe	-0.166	0.295	0.088	0.287	0.491	0.088	0.246	457.96%	179.55%	235.23%
prd_spe	0.061	0.841	0.835	0.805	0.774	0.835	0.858	-7.31%	2.75%	0.72%
subc_spe	0.155	0.379	0.388	0.339	0.289	0.388	0.401	-25.52%	3.35%	-2.32%
mod_spe	-0.020	0.109	0.012	0.094	0.178	0.012	0.092	1383.33%	666.67%	808.33%
fea_spe	0.197	0.546	0.388	0.446	0.506	0.388	0.556	30.41%	43.30%	40.72%
shp_spe	21.030	0.158	0.097	20.624	0.205	0.097	21.236	111.34%	21792.78%	62.89%

Table 3.17: Comparison of Misclassification Costs for Competitor Data (cost ratio 1:1)

Classifier	Misclassification Cost ($)			Cost Reduction by the Proposed Method	
	Proposed	OSS	Raw	Compared to OSS	Compared to Raw
Bayesian Network	1583.06	960.420	643.300	64.83%	146.08%
Decision Tree	1804.20	842.780	625.660	114.08%	188.37%
Logistic	1318.00	831.580	791.280	58.49%	66.57%
Naïve Bayes	821.28	709.980	628.920	15.68%	30.59%
Neural Network	1692.74	972.960	575.060	73.98%	194.36%
SVM	950.32	682.200	567.160	39.30%	67.56%

Table 3.18: Comparison of Misclassification Costs for Competitor Data (cost ratio 1:10)

Classifier	Misclassification Cost ($)			Cost Reduction by the Proposed Method	
	Proposed	OSS	Raw	Compared to OSS	Compared to Raw
Bayesian Network	2816.58	3912.24	4809.58	-28.01%	-41.44%
Decision Tree	2933.56	3107.54	5248.6	-5.60%	-44.11%
Logistic	3070.08	5695.18	7227.36	-46.09%	-57.52%
Naïve Bayes	3889.96	5779.86	4864.5	-32.70%	-20.03%
Neural Network	3233.66	3327	4647.38	-2.81%	-30.42%
SVM	3415.00	5207.76	5556.58	-34.42%	-38.54%

Table 3.19: Comparison of Misclassification Costs for Competitor Data (cost ratio 1:100)

Classifier	Misclassification Cost ($)			Cost Reduction by the Proposed Method	
	Proposed	OSS	Raw	Compared to OSS	Compared to Raw
Bayesian Network	14475.82	33430.44	46472.38	-56.70%	-68.85%
Decision Tree	14108.02	25755.14	51478	-45.22%	-72.59%
Logistic	28625.24	54331.18	71588.16	-47.31%	-60.01%
Naïve Bayes	34130.66	56478.66	47220.3	-39.57%	-27.72%
Neural Network	20505.86	26867.4	45370.58	-23.68%	-54.80%
SVM	27535.54	50463.36	55450.78	-45.43%	-50.34%

Table 3.20: Comparison of Mean Difference for Competitor Data

Web metrics	Raw			Remained (1:100)			Removed	Remained (Majority) - Minority	Removed - Minority	Raw (Majority) - Minority
	Overall	Majority	Minority	Overall	Majority	Minority				
sim_in	0.011	.0073071	.0465136	0.025	0.003	.0465136	0.008	-93.55%	-82.80%	-84.29%
sim_out	0.014	.0127735	.0218677	0.013	0.003	.0218677	0.014	-86.28%	-35.98%	-41.59%
sim_text	0.029	.0259482	.058979	0.039	0.020	.058979	0.027	-66.09%	-54.22%	-56.00%
count_news	5.545	3.883547	22.15959	11.637	1.113	22.15959	4.191	-94.98%	-81.09%	-82.48%
count_search	18776.420	15187.84	54652.33	30267.480	5882.625	54652.33	16222.070	-89.24%	-70.32%	-72.21%

Table 3.21: Comparison of Misclassification Cost Using Removed Majority Examples as Training Dataset

	Cost 1:1		Cost 1:10		Cost 1:100	
	RUS	proposed	RUS	proposed	RUS	proposed
BN	2824.90	2717.80	15958.90	15104.5	148385	129999.35
DT	2458.50	2447.90	18161.10	17592.9	175397	168936.90
Logistics	2440.40	2413.20	20115.80	19430.6	197137	197502.40
NB	2603.50	2594.40	19929.20	19560.3	193251	189503.20
NN	2501.30	2440.40	19337.50	18603.4	187689	185920.50
SVM	2530.50	2490.70	22746.80	21485.5	225309	201039.50

Table 3.22: Comparison of Misclassification Costs Under Different Class Distribution (cost ratio 1:10)

	5%		10%		15%		20%	
	RUS	proposed	RUS	proposed	RUS	proposed	RUS	proposed
BN	16742.4	15354.5	14542.3	13457.7	13554.7	13267.4	13522.1	13000
DT	14360.8	15727.2	13330.2	11710.5	14575.4	12654.7	12454.6	11454.5
Logistics	16756.6	13359	14226.6	13456.1	15365.8	14753.2	13654	12400.1
NB	21276.7	18953.7	19232.2	19266.5	18656.5	17468.6	17543.9	11535.3
NN	16868.1	15800.9	12859.1	15038.9	13545.6	14987.3	13565.5	11020.1
SVM	14445.2	13987.2	14645.5	14662.2	14656.4	13765.2	14263.6	12503.6

Table 3.23: Comparison of Misclassification Costs Under Different Class Distribution (cost ratio 1:100)

	5%		10%		15%		20%	
	RUS	proposed	RUS	proposed	RUS	proposed	RUS	proposed
BN	115113.3	105887.2	115113.3	105887.2	115113.3	105887.2	115113.3	105887.2
DT	107179.2	90327.5	107179.2	90327.5	107179.2	90327.5	107179.2	90327.5
Logistics	125647.6	117314	125647.6	117314	125647.6	117314	125647.6	117314
NB	187873	181508	187873	181508	187873	181508	187873	181508
NN	94191.1	98843.3	94191.1	98843.3	94191.1	98843.3	94191.1	98843.3
SVM	155739.5	100427	155739.5	100427	155739.5	100427	155739.5	100427

Table 3.24: Comparison of Misclassification Costs Between SMOTE and Proposed Method

	cost ratio 1:1		cost ratio 1:10		cost ratio 1:100	
	SMOTE	proposed	SMOTE	proposed	SMOTE	proposed
BN	3213.50	3317.80	13134.30	12656.00	123387.70	105887.20
DT	3277.86	3480.30	12040.80	11710.50	92323.80	90327.50
Logistics	3022.50	3037.90	13754.20	13426.10	123444.50	117314.00
NB	2453.60	2717.90	19302.30	18966.00	194543.30	181507.90
NN	3111.50	3141.00	17030.80	16038.90	103232.60	98843.30
SVM	3160.20	3069.00	12403.20	12362.20	123232.40	100427.00

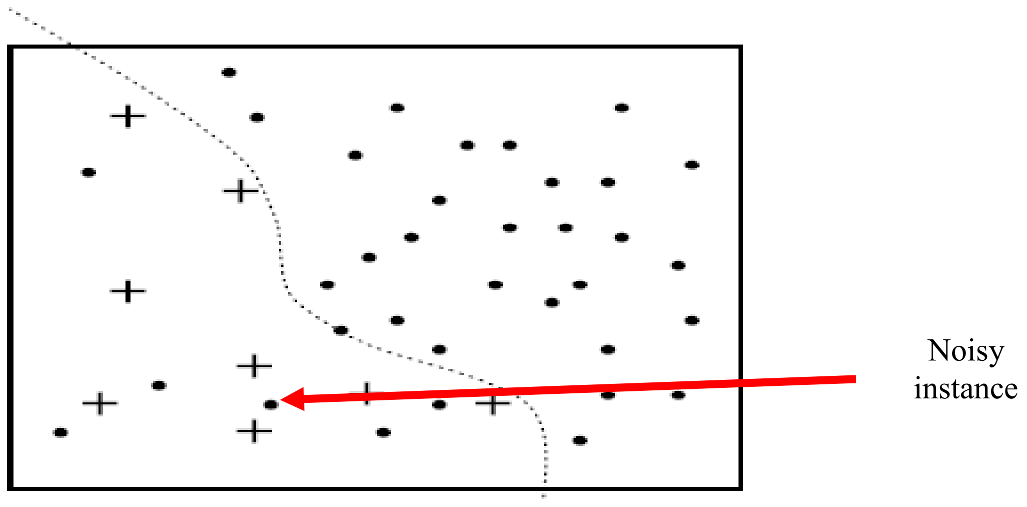


Figure 3.1: Illustration of Noisy Example

```
Input: a raw data set with N0 instances belonging to the majority class and N1 instances
belonging to the minority class;
    under-sampling ratio R;
    cost of misclassifying a majority instance as minority, c[majority][minority];
    cost of misclassifying a minority instance as majority, c[minority][ majority];
Output: a reduced data set with N0*(1-R) instances belonging to the majority class and
N1 instances belonging to the minority class
//initialization
For each instance A in the raw data set
        A.noise=0
End for
For each instance M in the raw data set
    find its nearest neighbor A
    If A.class = majority
            If A.class = M.class
                A.noise= A.noise - c[majority][minority]
            else
                A.noise = A.noise + c[minority][majority]
            End if
        End if
End for
remove from the raw data set N0*R majority instances with top noise levels
```

Figure 3.2: A Cost-oriented Under-sampling Method

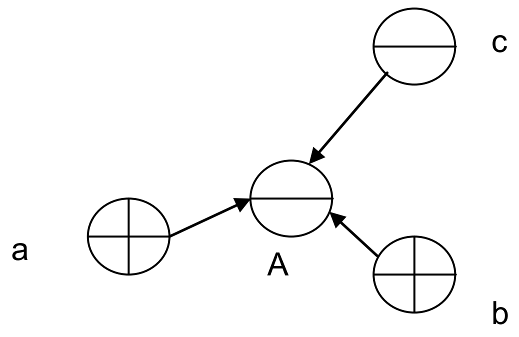


Figure 3.3: An Example of Data Location

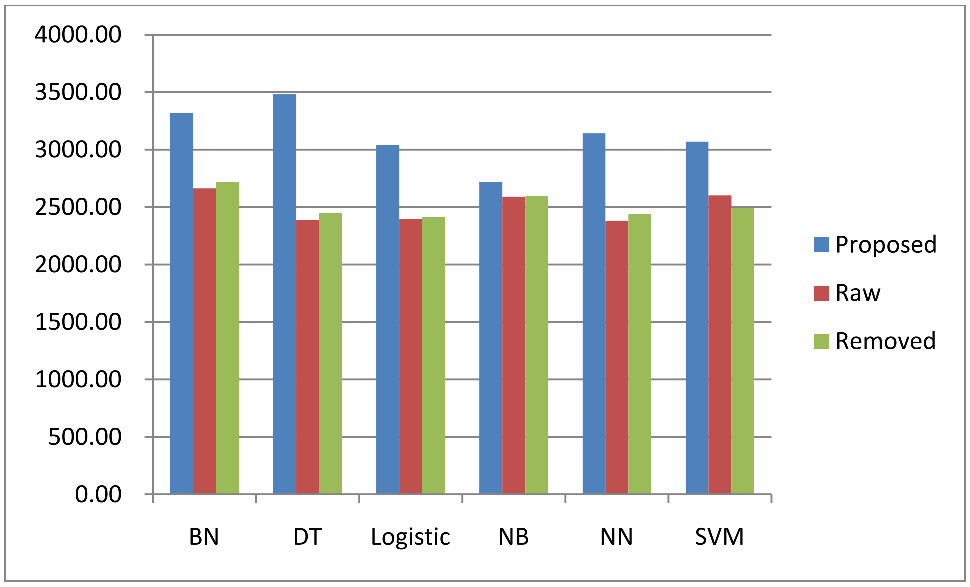


Figure 3.4: Misclassification Cost of Online Transaction Data (cost ratio 1:1)

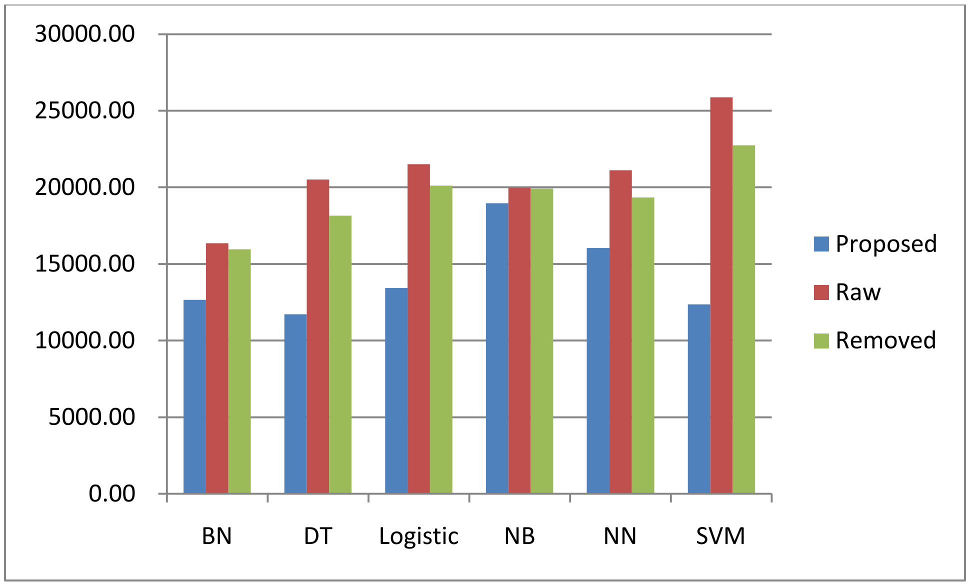


Figure 3.5: Misclassification Cost of Online Transaction Data (cost ratio 1:10)

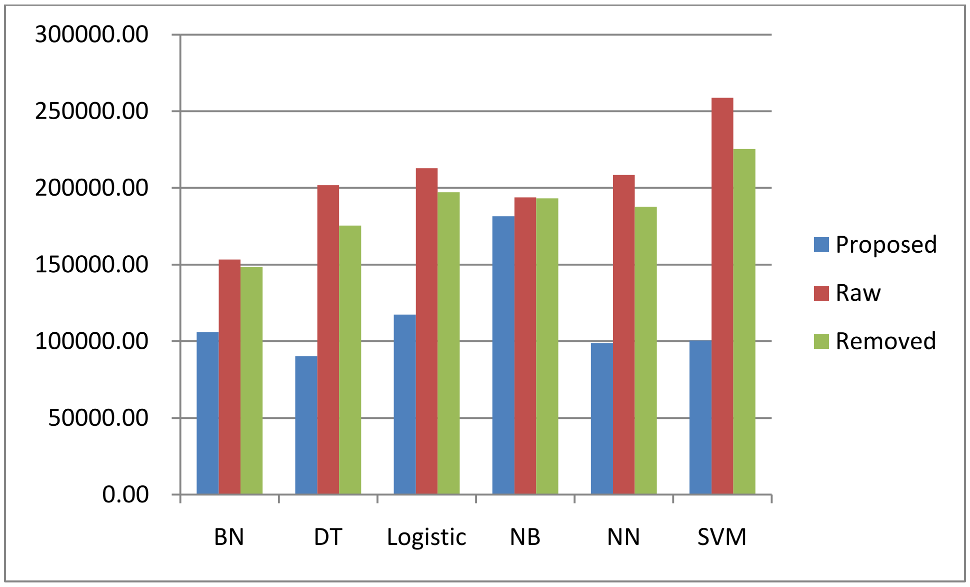


Figure 3.6: Misclassification Cost of Online Transaction Data (cost ratio 1:100)

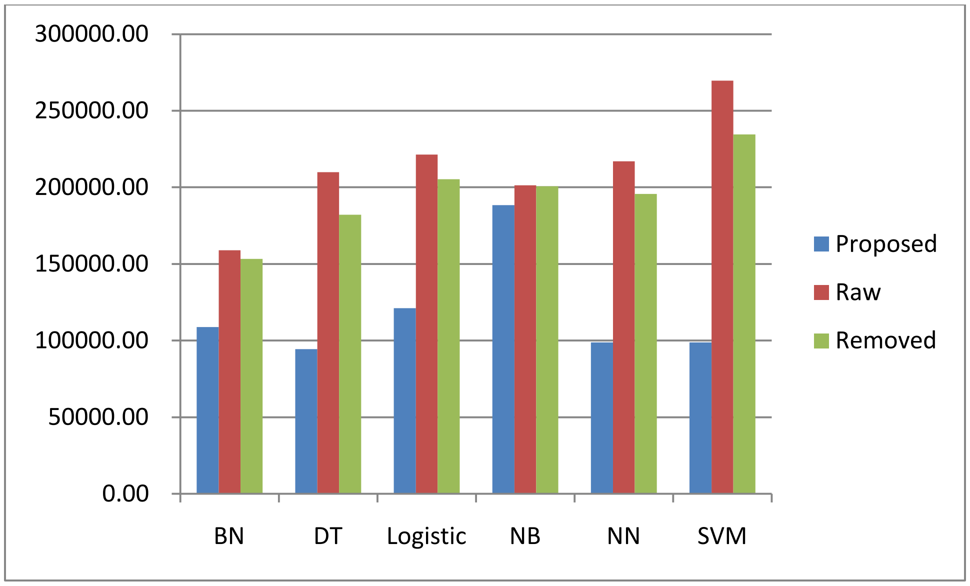


Figure 3.7: Misclassification Cost of Online Transaction Data (real cost ratio)

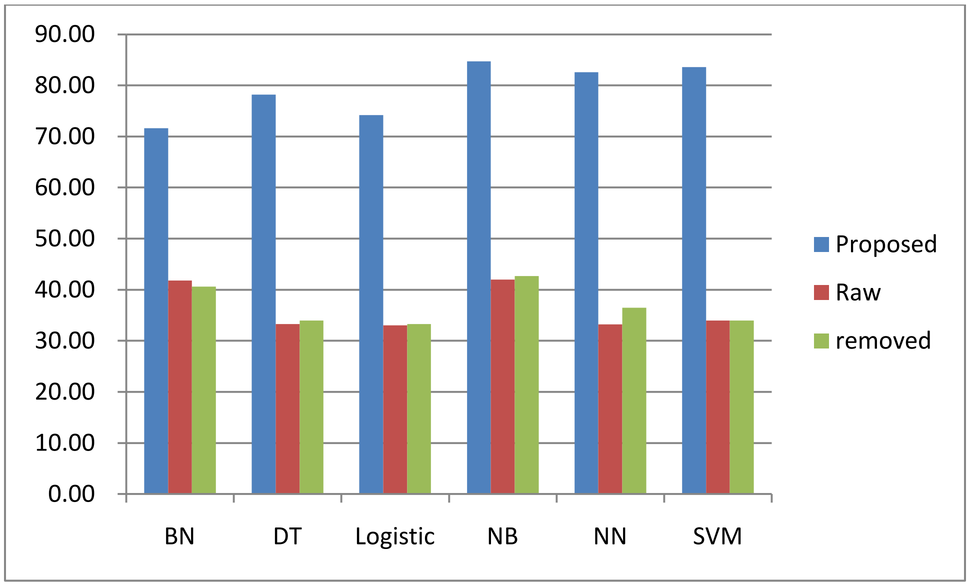


Figure 3.8: Misclassification Cost of Search Keyword Data (cost ratio 1:1)

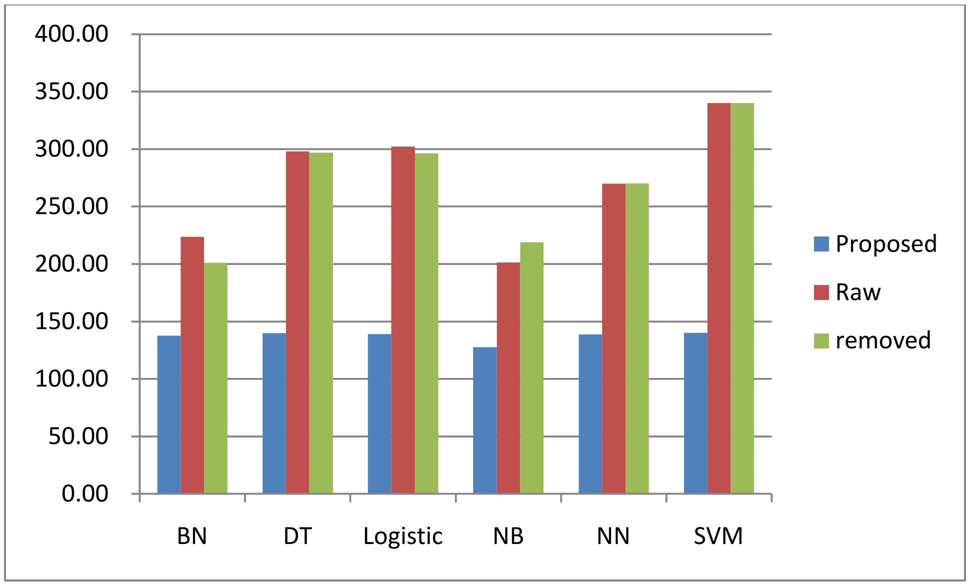


Figure 3.9: Misclassification Cost of Search Keyword Data (cost ratio 1:10)

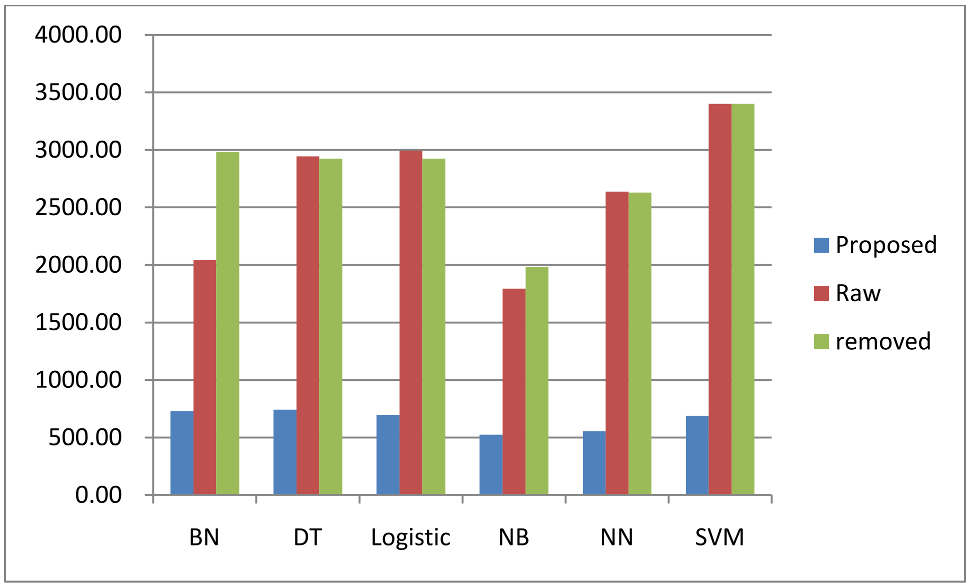


Figure 3.10: Misclassification Cost of Search Keyword Data (cost ratio 1:100)

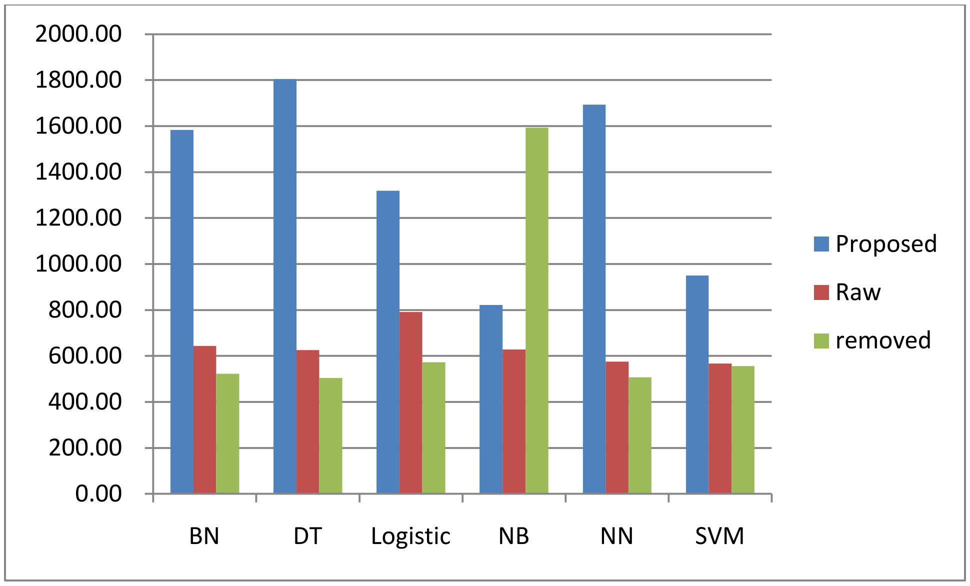


Figure 3.11: Misclassification Cost of Competitor Data (cost ratio 1:1)

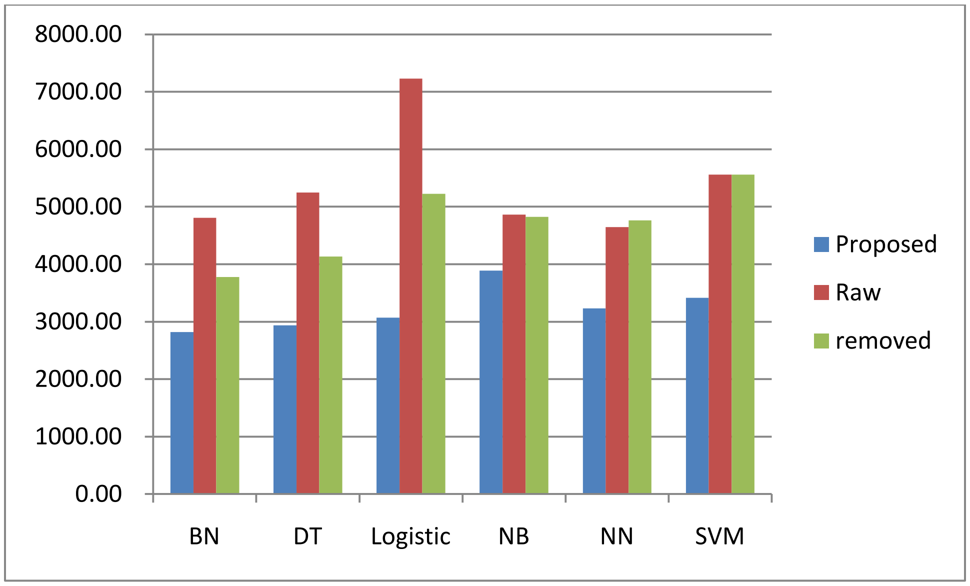


Figure 3.12: Misclassification Cost of Competitor Data (cost ratio 1:10)

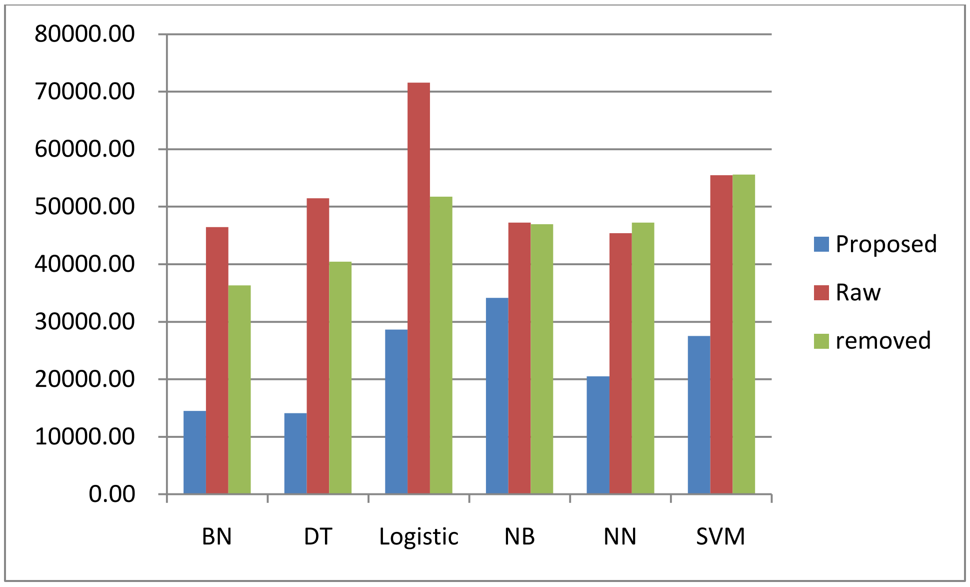


Figure 3.13: Misclassification Cost of Competitor Data (cost ratio 1:100)

CHAPTER 4

CONCLUSION AND FUTURE WORK

This dissertation develops two novel methods in cost-sensitive learning and imbalance learning. To the purpose of minimizing misclassification costs, these two methods incorporate the cost information into the learning process and achieve satisfying performance. This dissertation offers insights into these techniques and raises issues that merit future research. The purpose of this chapter is to summarize the findings of the dissertation and suggest ideas for the future research.

4.1 Conclusion and Contributions

In Chapter 2, I proposed an Iterative Cost-sensitive Naïve Bayes (ICSNB) that addresses the limitation of Naïve Bayes on probability estimation. Empirical tests using real-world data validate the proposed method and show that its performance outperforms other existing cost-sensitive learning methods. The consistent effect of the proposed method on cost reduction is demonstrated through the test results on multiple data sets varying in domains, sizes, class distribution, etc. Moreover, I tried five different cost matrixes to check the sensitivity of my method. Misclassification cost by ICSNB is always lower than other methods and there are only very few exceptional cases where the cost ratio is close to 1.

In Chapter 3, I developed a novel cost-based under-sampling method aiming to solve the ineffectiveness of current classifiers in imbalance data. I proposed to identify and remove noisy and costly majority instances to alleviate the impact of those instances on predicting minority samples. I tested my method through three real business case studies, i.e., online shopping, search engine marketing (SEM), and competitor identification. This empirical study provides support for my new under-sampling method in imbalance learning. Using removed samples to form a new training data, additional experiments further confirm that those removed instances are noisy instances and can cause higher costs in prediction. Moveover, experiment results shown the generaliability of the proposed method under different distribution levels.

My study makes the following contributions to the current literature:

- Proposes a new cost-sensitive learning method specified for Naïve Bayes classifier
- Reduces computation complexity in current Naïve Bayes research that focuses on improving probability accuracy
- Develops a novel intelligent under-sampling method
- Validates and compared the performance of cost-sensitive imbalance learning in business applications

4.2 Future Work

This dissertation only serves as the first step toward my efforts in cost-sensitive and imbalance learning. Future work needs to be extended to take care of some limitations and unsolved issues associated with this dissertation.

1. Distance measure is the one of key issues in the ICSNB method and it can directly affect the formation of the new training set and then influence the instance ranking. Although the Euclidean distance is the popular measurement for dissimilarity measure, for a data set with categorical attributes, it is interesting and helpful to explore some new and more effective distance measures.
2. As another key issue in the ICSNB method, the choice of optimal k, can also have a direct impact on the performance of the prediction model. The greedy search approach for optimal k in my study did not cost too much time and effort. But, it will be painful when dealing with a large data set, which is actually very common in our lives. Developing a rule to quickly derive an optimal k through analyzing the characteristics of input data can be a very interesting research topic for the future.
3. In the ICSNB method, hard instances are labeled using the normal Naïve Bayes approach. I analyzed the quantity of hard instances for each experimental data set and found that the number is usually less than 10. If those hard instances are the target instances and are assigned wrong labels, it will increase the overall cost and can result in a totally different result for a small data set. Thus, a future study focusing on those unsolved hard instances is needed.

4. Under-sampling is one of approaches for the imbalanced data. However, if we are facing small data, under-sampling will result in insufficient data points for learning. Therefore, extending my cost-based sampling method to over-sampling can be very useful to improve the current study.
5. Both of my proposed methods are designed to solve binary class problems. In the real world, there are many multiclass problems. Although my transformation method in the ICSNB is an alternative solution, the cost matrix is much more complex in multiclass data and cannot be easily converted into a 2X2 cost matrix. Therefore, developing a cost-sensitive method for multiclass problems is a direction for my future research.

REFERENCES

Abe, N., Zadrozny, B., and Langford, J. 2004. "An Iterative Method for Multi-class Cost-sensitive Learning," *KDD '04*, Seattle, Washington.

Aha, D. W., Kibler D., and Albert, M. C. 1991. "Instance-Based Learning Algorithm," *Machine Learning* (6), pp. 37-66.

Barandela, R., Valdovinos, R.M., Sanchez, J.S., & Ferri, F.J. 2004. "The Imbalanced Training Sample Problem: Under or Over Sampling?" *In Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition (SSPR/SPR'04), Lecture Notes in Computer Science 3138*, pp. 806-814.

Batista, G.E.A.P.A., Prati, R.C., and Monard, M.C. 2004. "A Study of the Behavior of Several Methods for Balancing Machine learning Training Data," *SIGKDD Explorations* 6(1), pp. 1-20.

Bauer, E., and Kohavi, R. 1999. "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants," *Machine Learning* (36:1-2), pp. 105-139.

Blake, C., Keogh, E., and Merz, C. J. 1999. UCI Repository of Machine Learning Databases. Department of Information and Computer Science, University of California at Irvine, CA. http://archive.ics.uci.edu/ml/

Breiman, L. 1996a. "Bagging Predictor," *Machine Learning* (24), pp. 123-140.

Breiman, L. 1996b. "Heuristics of Instability and Stabilization in Model Selection," *The Annals of Statistics* (24:6), pp. 2350-2383.

Cestnik, B. 1990. "Estimating Probabilities: A Crucial Task in Machine Learning," *In Proceedings of the Ninth European Conference on Artificial Intelligence*, Stockholm, Sweden: Pitman.

Cestnik, G., Kononenko, I., and Bratko, I. 1987. “ASSISTANT-86: A Knowledge Elicitation Tool for Sophisticated Users,” In Progress in *Machine Learning*, Eds. I. Bratko and N. Lavrac. Sigma Press.

Chai, X., Deng, L., Yang, Q., and Ling, C.X.. 2004. Test-Cost Sensitive Naïve Bayesian Classification. *In Proceedings of the Fourth IEEE International Conference on Data Mining*. Brighton, UK : IEEE Computer Society Press.

Chawla, N.V., Hall, L.O., Bowyer, K.W., & Kegelmeyer, W.P. 2002. "SMOTE: Synthetic Minority Oversampling Technique," *Journal of Artificial Intelligence Research* (16), pp. 321-357.

Cover, T.M., and Hart, P.E. 1967. “Nearest Neighbor Pattern Classification,” *IEEE Transaction on Information Theory* (13:1), pp. 21-27.

Cui, D., and Curry, D. 2005. “Prediction in Marketing Using the Support Vector Machine,” *Marketing Science* (24:4), pp. 595-615.

Domingos, P., and Pazzani, M. 1997. “On the Optimality of the Simple Bayesian Classifier under Zero-one Loss,” *Machine Learning* (29), pp. 103-130.

Domingos, P. 1999. “MetaCost: A General Method for Making Classifiers Cost-Sensitive,” *In Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, ACM Press, pp. 155-164.

Drummond, C., and Holte, R. C. 2000. “Exploiting the Cost (In)sensitivity of Decision Tree Splitting Criteria,” *In Proceedings of the Seventeenth International Conference on Machine Learning,* Stanford, CA.

Duda, R., and Hart, P. 1973. *Pattern Classification and Scene Analysis*, Wiley.

Elkan, C. 2001. "The Foundations of Cost-Sensitive Learning," *In Proceedings of the Seventeenth International Joint Conference of Artificial Intelligence*, Seattle, Washington: Morgan Kaufmann, pp. 973-978.

Fisher, R. A. 1936. “The Use of Multiple Measurements in Taxonomic Problems,” *Ann. Eugenics* (7), pp. 179-188.

Frank, E., Hall, M., and Pfahringer, B. 2003. “Locally Weighted Naive Bayes,” *in Proceedings of the Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, pp. 249-256.

Friedman J. H. 1997. "On Bias, Variance, 0/1 – Loss, and the Curse-of-Dimensionality," *Data Mining and Knowledge Discovery* (1), pp. 55-77.

Friedman, N., Geiger, D., and Goldszmidt, M. 1997. "Bayesian Network Classifiers," *Machine Leaning* (29:2-3), pp. 131-163.

Good, I. J. 1965. *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*, MIT Press.

Guo, H., and Viktor, H.L. 2004. "Learning from Imbalanced Data Sets with Boosting and Data Generation: The DataBoost-IM Approach," *ACM SIGKDD Explorations* (6:1), pp. 30-39.

Han, J., and Kamber, M. 2005. *Data Mining: Concepts and Techniques*, second edition, Morgan Kaufmann.

Hand, D. J., and Yu, K. 2001. "Idiot's Bayes – Not So Stupid After All?" *International Statistical Review* (69:3), pp. 385-398.

Hastie, T., Tibshirani, R., and Friedman, J. 2009. *The Elements of Statistical Learning*, Springer.

Hulse, J.V., Khoshgoftaar, T.M., and Napolitano, A. 2007. "Experimental Perspective on Learning from Imbalanced Data," *In Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR.

Japkowicz, N., and Stephen, S. 2002. "The Class Imbalance Problem: A Systematic Study," *Intelligent Data Analysis* (6:5), pp. 429-450.

Jen, L., Chou, C., and Allenby, G. M. 2003. "A Bayesian Approach to Modeling Purchase Frequency," *Marketing Letters* (14:1), pp. 5-20.

Jiang, L., Zhang, H., and Cai, Z. 2009. "A Novel Bayes Model: Hidden Naïve Bayes," *IEEE Transaction on Knowledge and Data Engineering* (21:10), pp. 1361-1371.

Kermanidis, K. L. 2009. "The Effect of Borderline Examples on Language Learning.," *Journal of Experimental & Theoretical Artificial Intelligence* (21:1), pp. 19-42.

Kim, Y., Street, W.N., Russell, G.J., and Menczer, F. 2005. "Customer Targeting: A Neural Network Approach Guided by Genetic Algorithms," *Management Science* (51:2), pp. 264-276.

King, G., and Zeng, L. 2000. "Logistic Regression in Rare Events Data," Working Paper.

Kubat, M., and Matwin, S. 1997. "Addressing the Curse of Imbalanced Data Sets: One-sided Sampling," *In Proceedings of the Fourteenth International Conference on Machine Learning,* Nashville, TN, pp. 179–186.

Kukar, M., and Kononenko, I. 1998. "Cost-Sensitive Learning with Neural Netwotks," *In the Proceedings of 13th European Conference on Artificial Intelligence,* Brighton, UK.

Lemmens, A., and Croux, C. 2006. "Bagging and Boosting Classification Trees to Predict Churn," *Journal of Marketing Research* (43:2), pp. 276-286.

Leroy, G., Miller, T., Rosemblat, G., and Browne, A. 2008. "A Balanced Approach to Health Information Evaluation: A Vocabulary-based Naïve Bayes Classifier and Readability Formulas," *Journal of The American Society for Information Science and Technology* (59:9), pp. 1409-1419.

Ling, C.X., Yang, Q., Wang, J., and Zhang, S. 2004. "Decision Trees with Minimal Costs," *In Proceedings of 2004 International Conference on Machine Learning (ICML'2004)*, Banff, Alberta, Canada.

Mani S., Shankle, W. R., Pazzani, M. J., Smyth, P., and Dick, M. B. 1997. "Differential Diagnosis of Dementia: A Knowledge Discovery and Data Mining (KDD) Approach," *In the Proceedings of AMIA Annual Fall Symposium*, Nashville, TN.

Melville, P., Saar-Tsechansky, M., Provost, F., and Mooney, R.J. 2004. "Active Feature Acquisition for Classifier Induction," *In Proceedings of the Fourth International Conference on Data Mining*. Brighton, UK.

Melville, P., Saar-Tsechansky, M., Provost, F., and Mooney, R.J. 2005. "Economical Active Feature-value Acquisition through Expected Utility Estimation," *UBDM Workshop, KDD 2005*.

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

Murphy, K.P. 2007. "Generative Classifiers," Research Notes.

Pant, G., and Sheng, O. R. L. 2009. "Avoiding the Blind Spots: Competitor Identification Using Web Text and Linkage Structure," In *Proceedings of 30th International Conference on Information Systems (ICIS),* Phoenix, AZ.

Pazzani, M., Muramatsu, J., and Billsus, D. 1996. "Syskil and Webert: Identifying Interesting Web Sites," *In Proceedings of the Thirteenth National Conference on Artificial Intelligence*, Portland, Oregon, pp. 54-61.

Provost, F. 2000. “Machine Learning from Imbalanced Data Sets 101,” Invited paper, in *Workshop on Learning from Imbalanced Data Sets*, AAAI, Texas, USA.

Provost, F., and Domingos, P. 2000. “Well-trained PETs: Improving Probability Estimation Trees,” CDER Working Paper #00-04-IS, Stern School of Business, New York University.

Provost, F., and Fawcett, T. 2001. “Robust Classification for Imprecise Environments,” *Machine Learning* (42:3), pp. 203-231.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers.

Rossi, P.E., McCulloch, R.E., and Allenby, G.M., 1996, “The Value of Purchase History Data in Target Marketing,” *Marketing Science* (15:4), pp. 321-340.

Rossi, P. E., and Allenby, G. M. 2003. “Bayesian Statistics and Marketing,” *Marketing Science* (22:3), pp. 304-328.

Sarkar, S., and Sriram, R. S. 2001. “Bayesian Models for Early Warning of Bank Failures,” *Management Science* (47:11), pp. 1457-1475.

Seewald, A. K. 2007. “An Evaluation of Naïve Bayes Variants in Content-based Learning for Spam Filtering,” *Intelligent Data Analysis* (11), pp. 497-524.

Scott, C. A., and Yalch, R. F. 1980. “Consumer Response to Initial Product Trial: A Bayesian Analysis,” *Journal of Consumer Research* (7), pp. 32-41.

Sismeiro, C., and Bucklin, R.E. 2004. "Modeling Purchase Behavior at an E-commerce Web site: A Task-completion Approach," *Journal of Marketing Research* (41:3), pp. 306-323.

Titterington, D. M., Murray, G. D., Murray, L. S., Spiegelhalter, D. J., Skene, A. M., Habbema, J. D. F., and Gelpke, G. J. 1981. “Comparison of Discrimination Techniques Applies to a Complex Data Set of Head Injured Patients,” *Journal of The Royal Statistical Society*, Series A, 144, pp. 145-175.

Tomek, I. 1976. "Two Modifications of CNN," *IEEE Transactions on Systems, Man and Communications,* SMC-6, pp.769-772.

Tong, S., and Koller, D. 2001. ”Support Vector Machine Active Learning with Applications to Text Classification,” *Journal of Machine Learning Research* (2:2), pp. 45–66.

Turney, P. D. 1995. "Cost-Sensitive Classification: Empirical Evaluation of a Hybrid Genetic Decision Tree Induction Algorithm," *Journal of Artificial Intelligence Research* 2:369-409.

Turney, P. D. 1997. "Cost-sensitive Learning Bibliography," Online Bibliography, Institute for Information Technology of the National Research Council of Canada, Ottawa, Canada, 1997.

Turney, P. D. 2000. "Types of Cost in Inductive Concept Learning," *Workshop on Cost-Sensitive Learning at the Seventeenth International Conference on Machine Learning (WCSL at ICML-2000)*, Stanford University, California.

Venkatesan, R., Kumar, V., and Bohling, T. 2007. "Optimal Customer Relationship Management Using Bayesian Decision Theory: An Application for Customer Selection," *Journal of Marketing Research* (44:4), pp. 579-594.

Viaene, S., Derrig, R., and Dedene, G. 2004. "A Case Study of Applying Boosting Naïve Bayes to Claim Fraud Diagnosis," *IEEE Transactions on Knowledge and Data Engineering* (16:5), pp. 612-620.

Weiss, G. M., McCarthy, K., and Zabar, B. 2007. "Cost-Sensitive Learning vs. Sampling: Which Is Best for Handling Unbalanced Classes with Unequal Error Costs?" *In Proceedings of the 2007 International Conference on Data Mining*, CSREA Press, pp. 35-41.

Weiss, G. M., and Provost, F. 2003. "Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction," *Journal of Artificial Intelligence Research* (4), pp. 211-255.

West, P. M., Brockett, P. L., and Golden, L. L. 1997. "A Comparative Analysis of Neural Networks and Statistical Methods for Predicting Consumer Choice," *Marketing Science* (16:4), pp. 370-391.

Yang, Q., Ling, C., Chai, X., and Pan, R. 2006. "Test-Cost Sensitive Classification on Data with Missing Values," *IEEE Transaction on Knowledge and Data Engineering* (18:5), pp. 626-638.

Yu, H., Han, J., and Chang, C. 2004. "PEBL: Web Page Classification without Negative Examples," *IEEE Transactions on Knowledge and Data Engineering* (16:1), pp. 1-12.

Zadrozny, B., and Elkan, C. 2001. "Learning and Making Decisions When Costs and Probabilities Are Both Unknown," *In the Proceedings of the Seventh International*

Conference on Knowledge Discovery and Data Mining (KDD'01), San Francisco, CA*,* pp. 204-213.

Zhang, H., and Su, J. 2008. “Naïve Bayes for Optimal Ranking,” *Journal of Experimental & Theoretical Artificial Intelligence* (20:2), pp. 79-93.

Zhou, Z., and Liu X. 2006. “Training Cost-sensitive Neural Networks with Methods Addressing the Class Imbalance Problem,” *IEEE Transaction on Knowledge and Data Engineering* (18:1), pp. 63-77.

Zhu, X., and Wu, X. 2005. “Cost-Constrained Data Acquisition for Intelligent Data Preparation,” *IEEE Transaction on Knowledge and Data Engineering* (17:11), pp. 1542-1556.