CHAPTER ONE

INTRODUCTION

Class imbalance is one of the challenges of machine learning and data mining fields. This chapter provides a gentle introduction to the problem of class imbalance and their difficulties that hinder the performance of algorithms. It also includes the objective and scope of this dissertation and finally the organization of chapters.

1.1 Introduction

Classification is one of the main data mining and machine learning tasks, which extracts useful information using intelligent methods. Given a data set with a set of objects each of them represented by a vector of attributes with one of the attributes as the class, the classification process is finding a model for class attribute as a function of the values of other attributes [**1**].

The classification problem of imbalanced data can appear in two types of data sets: two classes, where the instances of one class outnumber the instances of other and multi classes, where the applications have more than two classes. The multi class imbalance problem is an extension of the traditional two class imbalanced data where a data set consists of k classes instead of two. While imbalance is said to exist in the binary class imbalance problem when one class severely outnumbers the other class, when extended to multiple classes, the effects of imbalance are even more problematic. The class have overwhelmed called the majority class while the other is called minority class. However, in many applications the class has lower instances is the more interesting and important one. The imbalance problem heightens whenever the class of interest is relatively rare and has small number of instances compared to the majority class. Moreover, the cost of misclassifying the minority class is very high in comparison with the cost of misclassifying the majority class [**2**] for examples; consider cancer versus non-cancer or fraud versus un-fraud in first example the error of misclassification of positive class (cancer) as negative (non-cancer) is very big and may cause disaster or death. The same case in the second example, the error of misclassification of positive class (fraud) as negative (non-fraud) is very big and may cause huge losses.

The class imbalance can be intrinsic property or due to limitations to obtain data such as cost, privacy and large efforts [**3**].

Many real world applications suffer from these phenomena such as medical diagnosis, fraud detection (credit card, phone calls, insurance), network intrusion detection, pollution detection, fault monitoring, biomedical, bioinformatics and remote sensing (land mine, under water mine).As examples, both intrusion detection and fraud detection are considered as highly imbalance class problem. Although there is massive data however, most of them are legitimate and a little are intruders/fraudulent.

1.2 Problem Statement

Class imbalance is one of the challenges of machine learning and data mining fields. Imbalance data sets degrades the performance of data mining and machine learning techniques as the overall accuracy and decision making be biased to the majority class which lead to misclassifying the minority class samples or furthermore treated them as noise.

There are different difficulties caused by imbalance classes that hinder the performance of machine learning and data mining techniques [**4**]:

Firstly: the class distribution, the standard classifiers such as decision trees and neural networks assume that the training samples are equally distributed among classes. However, in many real applications the ratio of the minority class is very low (1:100, 1:1000 or may exceed 1 to 10000).

Secondly: lack of data, few samples of minority class in training set tends the classifiers to falsely detect them and the decision boundary be far from the true one.

Thirdly: concept complexity or overlapping, which refers to level of separability between data classes. High overlapped classes and high noise level produced higher complexity as depicted in Figure 1.1. Moreover, the discriminating rules can be difficult to induce if the examples of each class are overlapping at different levels in some feature space.

Fourthly: the existence of small disjuncts in a data set adds more complexity to the problem as depicted in Figure 1.2.

Fifthly: in most imbalance problems the cost of errors for different classes is uneven and usually it is unknown.

Furthermore, another problem associated with mining rare cases is reflected by the phrase: like a needle in a haystack in which the needle is obscured by a huge number of strands of hay that similarly to minority class samples that may be obscured by the majority class samples.

Various approaches have been proposed by the researchers for solving the imbalance problem. However, there is no general approach proper for all imbalance data sets and there is no unification frame work.

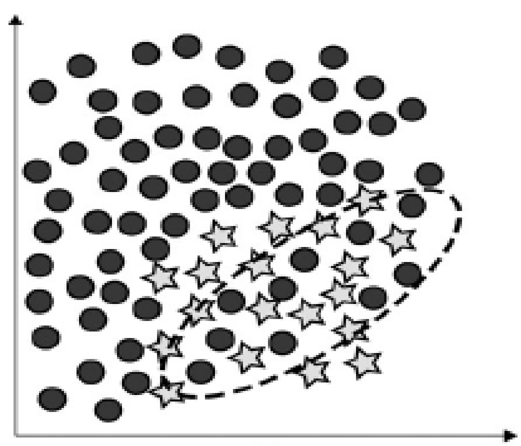

Figure 1.1: concept complexity (class overlapping) in imbalanced data

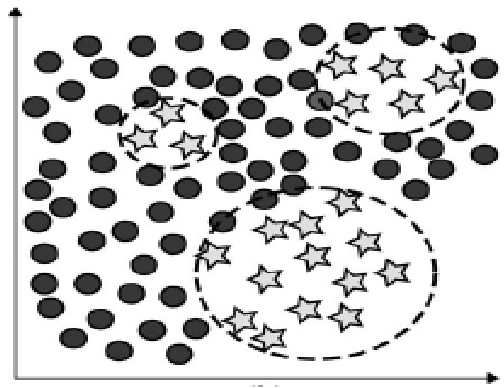

Figure 1.2: Small disjuncts in imbalanced data

1.3 Objectives

The major objective of this research is to investigate the highly imbalance classes problem through using an ensemble approach. Additional objectives are listed as follows:

- To investigate classifiers which are less sensitive to the class imbalance problem.
- To evaluate the classifiers performance under several circumstances.
- To improve the performance through using an ensemble approach.
- To compare the performance of different models on different highly imbalanced data.

1.4 Research Questions

- What are the best classifiers for dealing with highly imbalance data?
- What are the best ensemble (combination) approaches of classifiers for solution highly imbalance classes' problem?

1.5 Scope

- The scope of this research is limited for handling the imbalance class's data in:
  - Insurance fraud detection as two class imbalance problem
  - Network intrusion detection as multi class imbalance problem.
  - Other tested data sets:
    - Two class imbalanced data sets: German, Hepatitis, Haperman
    - Multi class imbalanced data sets: Thyroid, Lymphography, Glass, Landsat.

1.6 Thesis Structure

The rest of the dissertation is organized as follows:

Chapter two: provides a literature review for the given problem and all related works for both two and multi class imbalance problem.

Chapter three: includes experimental methodology and describe all methods and data sets used in our experiments.

Chapter four: includes the design and results discussion for all conducted experiments and the proposed approach for solving two-class imbalance problem.

Chapter five: includes the design and results discussion for all conducted experiments and the proposed approach for solving multi class imbalance problem.

Chapter six: provides conclusion and future works.

CHAPTER TWO

LITERATURE REVIEW

This chapter demonstrates the methodologies for handling imbalanced class problem for two class and multi class classification problems. Several methods proposed for solution the imbalance class problems include re-sampling and feature selection at the data level and other ones at the algorithm level such as cost sensitive and single class learning. However, most of these studies focused on two class classification problems and tried to improve the performance depending on the accuracy measure, which is an unsuitable performance measure for imbalanced data and few of them have been focused on multi class imbalanced classification problem. We start by introducing solutions for two class problem and then introduce solutions for multi class imbalanced data problem.

2.1 Imbalanced Two Class classification Review

2.1.1 Sampling based methods

Sampling methods is a preprocessing of data, which handle the imbalance problem by constructing balanced training data set and adjusting the prior distribution for minority and majority class [**3**] [**5**]. Sampling methods include under sampling and over sampling methods.

2.1.1.1 Undersampling

Under sampling balance the data by removing samples from majority class. Undersampling can be done using a non heuristic approach by randomly selecting samples from majority class [**6**] or using a heuristic approach such as Tomek link [**7**] , condensed NNR [**8**] or one sided selection [**9**].

Tomek link [**7**] defined pairs of samples belongs to different classes that are close to each other. They both take place in the decision boundary then tome links either removes the majority samples only or assume one of them representing noise. So Tomek links can be considered as undersampling method or as a cleaning method.

Condensed Nearest Neighbor (CNN) was proposed by Hart [**8**]. It applies a 1-NN (One-Nearest Neighbour) to select a consistent subset from the majority class by eliminate samples that are far from the decision border, by considering them less relevant and do not add additional information for learning.

One Sided Selection (OSS) [**9**] is an undersampling method that apply both Tomek links and CNN. Tomek links used to remove both noisy and borderline majority class samples. And CNN used to remove samples that are far from the decision border.

Li et al. [**10**] used granular support vector machines repetitive under sampling method (GSVM-RU). This method balances the majority class by extracting important samples and removing those unimportant ones. It significantly improved the efficiency of SVM model and reduced the computational cost.

Wilson's editing is an undersampling method proposed by Barandela et al. [**11**]. It employs the k-nearest neighbors for each majority sample and assigns its class based on 3-NN. If the sample is misclassified, it is excluded from the final dataset that represented the majority class.

A cluster based under sampling approach was proposed by Yen and Lee [**12**] to improve the classification accuracy for the minority class. They divided the training data into clusters and then selected the representative data for majority class samples from each cluster regarding the ratio of majority class samples to minority class samples. Their results showed that the cluster based under sampling improved prediction accuracy and it was more stable than other under sampling approach. Mostafizur et al. [**13**] modified the method proposed in [**12**] by separating the majority class into k clusters and selects a subset from each cluster. Then all subsets combined separately with the minority class to obtain k different training datasets. However, generally using under sampling may cause loss of useful information by removing significant patterns.

2.1.1.2 Oversampling

Oversampling balanced the data by creating copies of the existing samples or adding more samples to the minority class. Oversampling can be done using a non heuristic approach by randomly duplicating samples of minority class [**6**] or adding new samples using a heuristic approach. However, random over sampling may cause over fitting and may introduce additional computational tasks. To tackle this problem Chawla et al. [**14**] proposed a synthetic minority over sampling technique (SMOTE) by generating a synthetic examples rather than replacement with replication for the existing minority class samples. SMOTE works by selecting some or all the nearest neighbors for each minority sample and then take the difference between the feature vector for the minority sample under consideration and its nearest neighbor. Then, multiply this difference by a random number between 0 and 1 and add it to the feature vector under Consideration to produce the synthetic samples and add them to the minority class. This technique identifies more specific regions in the feature space for the minority class and so this makes the decision boundary for the minority class larger. The proposed technique maximized the performance of the classifier and biased the learning towards the minority class. However, it may lead to overlapping between classes because it does not take the neighbors of the minority class into consideration. Another drawback of SMOTE is appeared when the number of samples of minority class is not adequate for estimating the accurate probability distribution for the actual data.

SMOTE [**14**] generated a new samples for the minority class using the original ones in any further generation but Wei et al. [**15**] modified SMOTE and proposed a novel oversampling method called incremental SMOTE by considering the generated synthetic minority samples for further generation.

Borderline over sampling method was proposed in [**16**]. This method is similar to SMOTE, but it generated synthetic samples from the minority samples which take place around the borderline and they are most prone to be misclassified.

Juan and Li-li [**17**] proposed an oversampling method based on clustering and genetic algorithm. Firstly, k-means used to divide data into clusters and then the genetic algorithm used the parent samples in cluster to generate new minority samples applying mutation and crossover operations. GhaziKhan et al. [**18**] presented a wrapper based random oversampling that used genetic algorithm as the evolutionary optimization scheme to search the optimal regions for oversampling.

An investigation for the effects of imbalance ratio and the classifier was presented by Garcia et al. [**5**]. They evaluated several sampling methods RUS (random under sampling) and WE+ MSS (Wilson's editing with MSS condensing over the negative instances) as under sampling methods and SMOTE and gg-SMOTE (Gabriel- graph-based SMOTE). Their results showed that over sampling was outperformed under sampling in highly imbalance datasets as under sampling causes loss of significant patterns. The performance of evaluated methods (under sampling plus over sampling) was alike when the imbalance ratio was low.

Kamei et al. [**19**] evaluated the effects of four sampling methods (random over sampling, SMOTE, random undersampling and one sided selection) using four models (linear discrimination analysis, logistic regression analysis, neural network and classification tree). However, the sampling methods improved the prediction performance of linear and logistic models but there was no effect on neural network or classification tree performance.

.Kerdprasop and Kerdprasop [**20**] used Random over sampling and SMOTE to improve the performance of the learned model using decision tree induction, regression analysis, neural network and SVM. The highest sensitivity model given by random over sampling while SMOTE gives the highest specificity model. Moreover, they applied a cluster based feature selection which added a significant improvement to the predicting accuracy for the learned models.

Ramentol et al. [**21**] introduced a hybrid sampling method by integrate SMOTE with fuzzy rough set theory (FRST). Using this method improved SMOTE performance by eliminating the synthetic minority class samples which they had lower degree to the fuzzy region. While, recently a fuzzy distance based undersampling (FDUS) is combined with SMOTE in [**22**]. The combination of SMOTE and FRST performance have surpassed other SMOTE approaches.

A hybrid under sampling technique for mining unbalanced datasets was proposed by Ravi and Vasu [**23**]. They employed KRNN (k-reverse nearest neighbour) to detect the outliers and K-means clustering on the majority class. The proposed method was tested using several classifiers such as SVM, logistic regression (LR), radial basis function network (RBF), genetic programming and decision tree (J48). Their results showed that the proposed under sampling technique increased the classifier's performance.

Although SMOTE has advantages for balancing the data effectively however, it may bring noise. To overcome this problem, recently Mi [**24**] proposed an active learning SMOTE that selects the best and valuable samples for learning. The author introduced SVM into adapted SMOTE learning frame. Their results showed that the proposed method outperformed other learning models using SMOTE, undersampling and AdaBoost.

Also, a hybrid feature selection method was proposed in [**25**] by combining re-sampling and feature subset approaches. They used SMOTE for re-sampling, consistency subset evaluation method and genetic search for finding the optimal feature space and removing irrelevant features. The proposed method improved the classifier performance and outperformed the other feature selection methods. Although most of the learners benefits from sampling techniques, but the performance of sampling techniques depend on the dataset size and imbalance ratio [**26**].

Alibeigi et al. [**27**] presented an unsupervised feature selection for highly imbalanced data. The proposed feature selection method selects the more important and informative features regarding the minority class and remove the redundant features according to their probability density function.

Cuaya et al. [**28**] proposed a minority class feature selection called Feature Selection for Minority Class (FSMC). FSMC is a filter method that selects features whose minority class values significantly different from majority class values. This method return fewer attributes in less time in comparison with other feature selection methods.

A hybrid approach that combined feature selection with sampling is proposed in [**29**], [**30**], [**31**]. A wrapper-based feature selection was used with random undersampling. They also combined feature selection with sampling but in a repetitive manner by aggregating results obtained during each repetitive process. Authors in [**32**], [**33**] proposed a hybrid approach using subset filtering. The majority class divided into multiple subsets and then those subsets combined with the minority subsets to form balanced sets. The number of subsets depends on the imbalance ratio. They used Correlation based Feature Subset (CFS) to reduce the effect of imbalanced classes.

Authors in [**34**], [**35**] proposed an undersampling method based on feature selection. It eliminated weak and noise samples and select the strong samples for each specific feature using filter method.

2.1.3 Cost sensitive learning based methods

In many imbalance class problems, not only the data distribution is skewed but also the misclassification error cost is uneven. The cost learning techniques take the misclassification cost in its account by assigning higher cost of misclassification to the positive class (minority class) i.e. C(+,-)> C(-,+) and generate the model with lowest cost [**36**]. However, the misclassification errors costs are often unknown and furthermore, cost sensitive learning may lead to over fitting.

Another cost sensitive learning approach used in unbalance dataset is adjusting the decision threshold of the standard machine learning algorithms, wherever the selection of threshold is an effective factor on the performance of learning algorithms [**37**]. This approach moves the output threshold towards the inexpensive class (majority class) so as the samples with higher costs became harder to be misclassified.

Thach et al. [**38**] proposed accuracy- based learning (XCS) with cost sensitive. They identified a constraint reward function which maximizes the total reward of the positive class samples and improves the performance of XCS in imbalanced data. Alejo et al. [**39**] proposed a hybrid method based on Gabriel graphs technique and modified back propagation algorithm. They proposed new cost function based on minimum square error (MSE).

Authors in [**40**] investigated the performance of cost sensitive classifier regarding data with/without imbalanced classes. They used instance-weighting- based cost sensitive C4.5. From their experiments they recommended to use the normal distribution of data if the costs do not differ seriously and using balanced classes if the costs differ seriously. In [**41**] Sun et al. used instance-weighting method that assigns different error classification costs to positive and negative training samples.

Hong et al. [**42**] presented a hybrid kernel algorithm that combined SMOTE and Particle Swarm Optimization (PSO) and Radial Basis Function (RBF). OFS is employed to construct RBF classifier and PSO was used to determine the parameters of RBF.

Daneshmandi and Ahmadzadeh [**43**] proposed a hybrid approach that integrates a supervised learning using neural networks with an unsupervised learning using k-nearest neighbors clustering. Fuzzy C-means clustering (FCM) algorithm used to improve the performance of SVM on imbalanced data classes. FCM builds n clusters randomly and the membership for each training sample is calculated.

Adam et al. [**44**] improved the performance of the neural networks in imbalanced data using Particle Swarm Optimization. Back propagation learning used to train the neural network then PSO was applied to optimize the decision boundary for the output layer in trained neural network.

Bahnsen et al. [**45**] integrated sampling techniques and cost sensitive learning into improved random forests. They integrate sampling techniques into balanced random forests and cost sensitive learning into weighted random forests. In [**46**] Kothandan investigated the competence of using SMOTE sampling and cost sensitive learning with two step SVM. Ling and Sheng [**47**] proposed two empirical methods to deal with class imbalance. The first method combined cost sensitive learning with sampling and the second used cost sensitive learning by optimizing the cost ratio. They concluded that cost matrix have reduced when using the first method however, the second one have better performance.

Uyar et al. [**48**] examined the classification performance when using oversampling, under sampling and adjusting the decision threshold. Their results showed that the optimum true positive rate and false positive rate can be improved easily by adjusting the decision threshold. Also, Yan et al. [**49**] proposed an adjustment method for threshold based on Fisher discrimination.

2.1.4 Recognition based methods

In recognition based method or (one-class learning) the classifier learned on the just target class samples. This approach improves the performance of the classifier on unseen data by recognized only those belong to that class. Raskutti and Kowalczyk [**50**] investigated the effect of sampling, and weighted learning of a single class. They concluded that one- class learning can be a robust technique when dealing with imbalanced data and highly dimensional noisy feature space. Recently, A hybrid approach that combined undersampling with one class SVM (OSVM) was proposed by Kim and Ahn [**51**]. They used k-reverse nearest neighbors to remove outliers and OSVM is used to extract support vectors in the majority class. For solving parameter selection problem in imbalanced data, in [**52**] Zhuang and Dai used one-class learning to train on the minority class and then an optimization criteria is set using the generalization performance which is estimated from both minority and majority classes. One-class learning can perform better under certain conditions such as high dimensional data, however, many classifiers such as decision trees and Naive Bayes cannot be built by one class learning.

2.1.5 Ensemble- based Methods

Ensemble is a combination of multiple classifiers so as to improve the generalization ability and increase the prediction accuracy. The most popular combining techniques are boosting and bagging. In boosting, each classifier is dependent on the previous one, and focuses on the previous one's errors. Examples that are misclassified in previous classifiers are chosen more often or weighted more heavily. Whereas, in bagging, each model in the ensemble votes with an equal weight. In order to promote model variance, bagging trains each model in the ensemble using a randomly drawn subset of the training set [**53**].

Kang and Cho [**54**] proposed an ensemble of under sampled SVM (EUS SVMs). They integrated the good generalization ability of SVM by boosting ensemble scheme. Their proposed method overcame the drawback of under sampling method and reduced the time complexity of oversampling method.

Zhang and Wang [**55**] presented an ensemble model that combining cost sensitive SVM and query by committee (QBC) with AdaBoost learning. The majority class divided into several subsets regarding to imbalance ratio. Then QBC which is an active learning method is used to generate nominee training samples and selects the effective ones. AdaBoost is used to train the sub classifiers.

Khoshgoftaar et al. [**56**] studied empirically the use of different data sampling with Boosting including random undersampling, random oversampling, SMOTE, Borderline SMOTE and wilson's editing. The best performance usually obtained by undersampling. SMOTE and borderline SMOTE given better results than the random oversampling and wilson's editing. They concluded that Boosting improve performance over sampling methods.

A hybrid kernel machine ensemble by integrating two types of kernel machine one class SVM and binary SVM (BSVM) is proposed in [**57**]. Authors in [**58**]- [**59**] proposed a hybrid approach using random undersampling with AdaBoost (RUSBoost). They obtained the desired distribution by randomly remove samples from majority class. RUSBoost is simpler and faster technique comparing to SMOTEBoost and other technique.. Yuan and Ma in [**60**] improved the performance by using SMOTE with AdaBoost and an objective function using optimization technique such as genetic algorithm.

Xiong et al. [**61**] proposed an ensemble model that integrates sampling with AdaBoost using Naïve Bayes (NB) and decision tree C4.5 as base classifiers. Random sampling used with NB to denotes the data distribution and Undersampling used with C4.5/C4.5+AdaBoost.

To tackle the deficiency of undersampling, Liu et al. [**62**] proposed two ensemble models called Easy ensemble and balanced cascade ensemble. In easy ensemble the combined classifiers are trained on different subsets separately. In balanced cascade ensemble the combined classifiers trained sequentially using a guide in the sampling process for each classifier by removing samples those are classified correctly. Recently, Tianyu [**63**] used easy ensemble based feature selection. To improve the performance, PSO is applied to get the optimal feature subsets.

Khoshgoftaaret al. [**64**] proposed a filter-based feature ranking techniques. They applied an iterative feature selection strategy and combined it with sampling and boosting techniques. This method repeatedly employed data sampling followed by feaure selection. The ranked features sets taken from each iteration and applied to boosting learners.

An investigation on the performance of random sampling and advanced under sampling (CUBE) and two modeling techniques (gradient boosting and weighted random forests) was introduced by Burez and Poel [**65**]. They concluded that under sampling improved the prediction accuracy comparably with sophisticated under sampling which had no any effect on the performance. Also, they found that Boosting is a robust classifier but not surpassed the other techniques and Weighted random forest performed better than random forest.

Gue and Viktor [**66**] proposed an ensemble based learning approach (DataBoost-IM) that combined boosting with data generation. The hard examples were identified then they were used to generate synthetic examples for both classes to be focus by the next classifier component in the boosting procedure. However, synthetic examples prevented boosting from over fitting on hard examples. Another ensemble in a hierarchical frame was proposed by Zhang and Luo [**67**]. They proposed a parallel classification method to improve classifying speed; two classifiers (simple one and complicated one) were trained serially but worked in parallel. The results showed that their proposed approach effectively improved performance and speed.

An approach based on repeated sub-sampling was proposed by Khalilia et al. [**68**]. They compared the performance of SVM, bagging, boosting and Random Forest (RF). They emphasized the effectiveness of repeated sub-sampling in dealing with highly imbalance data sets. However, RF outperformed other methods plus its ability to estimate the importance of each variable in classification process.

Recently, a hybrid ensemble model that integrated sampling, clustering and bagging was proposed by Wang [**69**]. Firstly, the borderline majority samples were removed using Tome links undersampling technique. Then the remaining majority class divided into a number of subsets (clusters). These subsets were combined with the minority class using bagging learning technique. For diversity they used random forests and decision tree as base classifiers for the ensemble. Krawczyk et al. [**70**] proposed a hybrid ensemble model that combined feature selection with cost sensitive learning. Random feature subspaces are used for the diversity of ensemble. Cost matrix is used to construct the base classifiers. To promote the performance, an evolutionary algorithm was used for classifier selection and assignment of committee member weights.

Table [2] summarizes the advantages and drawbacks of the proposed methods for dealing with imbalance problem: resampling, cost sensitive learning, one class learning and ensemble approaches.

Table 2.1. The advantages and drawbacks of the proposed methods for dealing with imbalance problem

<table>
<tr><th>Method</th><th>Advantages</th><th>Limitations</th></tr>
<tr><td>Under-sampling</td><td rowspan="2">• Independent on underlying classifier.
• Can be easily implemented</td><td>• May remove significant patterns and cause loss of useful information</td></tr>
<tr><td>Over-sampling</td><td>• Time consuming: Introduce additional computational cost
• May lead to over-fitting</td></tr>
<tr><td>Cost sensitive</td><td>• Minimize the cost of misclassification (by biasing the classifier toward the minority class)</td><td>• The misclassification costs (the actual cost of errors) often are unknown</td></tr>
<tr><td>Recognition based</td><td>• Have better performance especially on high dimensional data</td><td>• Many classifiers such as decision trees and Naive Bayes cannot be built by one class learning.</td></tr>
<tr><td>Ensemble</td><td>• Better classification performance than individual classifiers
• Less likely to overfit
• More resilience to noise</td><td>• Time consuming</td></tr>
</table>

2.2 Imbalanced Multi Class classification Review

In the previous Sections, we introduced the related works and proposed solution for handling two classes' imbalanced data, however, in comparison with solutions of multi class imbalanced data which is an extension for the two class imbalanced problem and more complicated, we found that there are few focused and concentrated solutions for handling it. In addition to that, most of those proposed solutions do not directly apply in multi class classification problems.

Multi class classification problem can be solved using two different strategies: problem adaptation and problem decomposition. Problem adaptation solution carried out by directly manipulate and adapted specific algorithm using single classifier with multiple outputs. Whereas, the other solution accomplished by transforming and decomposing the multi class problem into small binary sub problems which referred as multi binary classification. Ensemble is made of those binary classifiers and the final output of the ensemble is a combination of binary classifiers outputs. There are several approaches for multi binary classification that differ on the used decomposition and integration techniques. The most popular ones are One against All, One against One and Error Correcting Outputs code [**71**]- [**72**].

For solving multi class classification in imbalanced data, a hybrid approach by combining clustering and sampling was proposed by Al-Roby and El-Halees [**73**]. The training data are divided into number of clusters and in each cluster SMOTE over sampling method has been applied. The new balanced data obtained by combining all clusters. Wang and Ou [**74**] proposed a solution that based into decomposition of multi class into binary sub problems. They developed a hierarchy multi class method One-Against-Higher-Order (OAHO). For k class, k-1 classifiers are built in a hierarchy regarding the order; i.e. if any sample misclassified by the first classifier it doesn't corrected by any lower classifier. Also, Jeatrakuland Wong [**75**] developed One-Against-All with Data Balancing (OAA-DB) algorithm that combined multi-binary classification and data balancing. It applied One-Against-All (OAA) multi-binary classification technique and data balancing technique that integrated undersampling using Complementary Neural Network and SMOTE oversampling method.

Recently, Lavanya et al. [**76**] proposed an approach that based on one against one (OAO) decomposition technique for multi class data. After the problem transformed into binary sub problems, they applied undersampling technique in each sub binary problem using ant colony optimization algorithm to extract the best subset.

To reduce the computational cost of sample selection Alaouiet al. [**77**] presented a clustering method based on Sample Selection (SS). SS is an undersampling method that selects the important majority samples from the critical clusters only (samples which are not close to border).

Prachuabsupakijet al. [**78**] proposed C-MEIN (Clustering with Sampling for Multi class Imbalanced using Ensemble) which is a hybrid ensemble that integrated clustering with sampling. The data samples are divided into two clusters using K-means algorithm. Then, in each cluster two resampling methods were used to balance data. The new balanced data are used to construct the ensemble.

Leung et al. [**79**] investigated the performance of combining feature subsets methods (Principle Component Analysis, Genetic Algorithm, Rough Set) with three sampling methods (undersampling, oversampling, undersampling+oversampling). The optimal pairs of feature subsets and sampling techniques obtained using Genetic algorithm.

Rafiahet al. [**80**] combined features extraction, sampling with Boosting. They extracted the important features using feature extraction algorithm, then SMOTE+Boosting applied to the extracted generated features.

Authors in [**81**], [**82**] proposed cost sensitive approach using SVM with ramp loss function to handle multi class classification. They developed an objective function that optimize performance measure such as g-mean, F-measure.

Very recent investigation proposed by Sainin et al. [**83**] have studied the use of random undersampling with replacement (SWR) with new proposed direct ensemble classifier for imbalanced multi class learning (DECIML). DECIMIL is a combination of two ensemble, the first one combined NB and 1- nearest neighbor and second one combined NB and k- nearest neighbors. However, SWR degraded F-measure in some datasets.

From the previous study we can conclude that various approaches have been proposed by the researchers for solving the imbalance problem. However, there is no general approach proper for all imbalance data sets and there is no unification frame work.

In addition to, most of these proposed methods try to improve performance and to obtain satisfactory results in term of accuracy. However, through this investigation new hybrid ensemble models will be proposed to handle imbalance class problem in order to improve the true detection rates per class and reduces the false alarms for both minority and majority classes.

2.3 Summary

This chapter gave an overview of major existing techniques related to imbalanced class problem, which is proposed for handling two-class and multi-class imbalance problem. The next chapters shall present proposed approaches for solving this problem through using hybrid ensemble approaches.

CHAPTER THREE

EXPERIMENTAL METHODOLOGIES

This chapter introduces the experimental methodology utilized in our experiments using different machine learning and data mining classifiers and different imbalanced data sets and the WEKA machine learning tool. Section 3.1, explains information about the data sets used for experiments. Section 3.2, provides details of sampling methods. Section 3.3 provides the details about the bases classification learners. Section 3.4 explains information about the Meta learning methods and finally Section 3.5 describes the performance measures used in evaluation.

3.1 Datasets

In total, nine real world imbalanced datasets were used for the experiments with different sizes and imbalanced ratios four of them are two class imbalanced data sets and the others are multi class imbalanced dataset.

3.1.1 Two Class Imbalanced Data

(1) Insurance Fraud detection data set

This dataset related to insurance fraud detection, which encompasses information about automobile insurance claims. It consists of 32 variables, 31 predictor variables and one class variable. The total of samples is 15,420 samples, 14,497 are non-fraudulent and 923 are fraudulent, which indicate the data set is highly imbalanced [**23**].

(2) German Dataset

This dataset obtained from the University of California at Irvine (UCI) repository, which is a contribution by Hans Hofmann. It is concerned regarding credit card applications. The purpose of the German credit data set is to predict whether a loan application is a good or a bad credit risk. The total of samples is 1000 samples, Number of attributes is 20, 700 are negative and 300 are positive [**84**].

(3) Hepatitis Dataset

This dataset obtained from the UCI repository. It is used to diagnose whether a hepatitis patient will die or live. The total of samples is 155, 123 LIVE and 32 DIE samples number of attributes is 20 [**85**].

(4) Haberman's Survival Dataset

This dataset obtained from the UCI repository. It contains the survival status of the patients who had undergone breast cancer surgery. The dataset has 306 samples each of which has 3 attributes. For this data 225 patients survived more than 5 years post surgery and 81 patients died within 5 years. The goal for this data is to predict the class (dead or alive) using the 3 input variables [**86**].

3.1.2 Multi Class Imbalanced Data

(1) Intrusion detection dataset

Intrusion detection dataset were prepared by MIT Lincoln Lab [**87**]. This Dataset consists of 41 attributes and one class label. 24 attack types classified into four main classes: Dos (Denial of Service), R2L (Unauthorized Access from a Remote Machine), U2R (Unauthorized Access to Local Super User (root)) and Probing. The data is highly imbalance, the training set divided as follows: 1000 are normal data, 1000 are probe, 3002 are Dos, 27 are U2R and 563 are U2L.

(2) Glass dataset

Glass Identification dataset was generated to help in criminological investigation. At the scene of the crime, the glass left can be used as evidence, but only if it is correctly identified. It contains 214 instances, 9 numeric attributes and class attribute. Each instance has one of 7 possible classes. The data distributed as 70, 76, 17, 13, 9 and 29 for Building Windows Float Processed Glass, Vehicle Windows Float Processed Glass, Building Windows Non-Float Processed Glass, Vehicle Windows Non-Float Processed Glass, Containers Non-Window Glass, Tableware Non-Window Glass and Headlamps Non-Window Glass class [**88**].

(3) Thyroid dataset

This dataset is used to determine whether a patient referred to the clinic has hypothyroid. The total number of instances is 3772, 21 attributes (15 binary, 6 continuous); three classes named: primary hypothyroid, compensated hypothyroid, normallyeach class has 93, 191 and 3488 instances respectively [**89**].

(4) Landsat Satellite image dataset

This dataset was generated from Landsat Multi-Spectral Scanner image data. It consists of the multi-spectral values of pixels in 3x3 neighbourhoods in a satellite image, and the classification associated with the central pixel in each neighbourhood. The aim is to predict this classification, given the multi-spectral values. In the sample database, the class of a pixel is coded as a number. It has 6 decision classes: 1, 2, 3, 4, 5 and 7 (class 6 has been removed because of doubts about the validity of this class) [**90**].

(5) Lymphography dataset

This lymphography domain is one of three domains provided by the Oncology Institute, University Medical Centre that has repeatedly appeared in the machine learning literature. It contains 148 total number of instances with 19 number of attributes including the class attribute. It has 4 classes distributed as 2 normal find, 81 metastases, 61 malign lymph and 4 fibrosis [**91**].

3.2. Sampling Methods

Sampling methods is a preprocessing of data, which handle the imbalance problem by constructing balanced training data set and adjusting the prior distribution for minority and majority class. In our experiments, we employ under sampling and oversampling as sampling methods to modify and balance our imbalanced two class data sets.

(1) Undersampling

In our experiments we use random Under-Sampling (RUS), which is a non-heuristic method that aims to balance class distribution through the random elimination of majority class instances to get a set that equals to the minority class.

(2) Oversampling

To oversampled our data, we use SMOTE [**14**]. SMOTE increase the number of minority class by taking the difference between a feature vector (minority class sample) and one of its k nearest neighbors (minority class samples). And multiply this difference by a random number between 0 and 1. Then add this difference to the feature value of the original feature vector to create a new feature vector (See figure 3.1)

```
Add newminority class instance by:
    for each minority class instancec
    –neighbours = GetKNN (k)
    –n = Randompickonefromneighbours
    –Create a new minority class r instance using c′s feature vector by:
  r.features = c.features + (c.features – n.features) ∗ rand(0,1)
```

Figure 3.1 Algorithm of SMOTE

The above steps are applied when the features are continuous. For the nominal features SMOTE takes majority vote between the feature vector under consideration and its k nearest neighbors for the nominal feature value (In the case of a tie, choose at random) and assign that value to the new synthetic minority class sample.

3.3 The Basic Learners

In our experiments, we use seven distinct classifiers from the top and most used data mining and machine learning algorithms including Naïve Bays, Support Vector Machine, Back propagation neural network, radial basis function network, the decision trees C4.5, Random Forests and Random Trees.

(1) Naïve Bays

Naïve Bays (NB) is a simplified Bayesian probability theory, which is considered as a learning method as well as a statistical method for classification. It calculates explicit probabilities for hypothesis and it is robust to noise in input data. NB based on the conditional assumption that variables are independent within each output label. Although its conditional assumption is rarely true in real world application, it is distinguished by its surprising and competitive performance in classification [**92**].

(2) Support Vector Machine

Support Vector Machine (SVM) is one of the robust, popular and successful classification algorithms, which based on the risk minimization. It constructs hyper planes in a high dimensional space and finding the optimal one that maximizes the margin of training data. The basic SVM supports only binary classification, but extensions have been proposed to handle the multiclass classification case as Well as additional parameters and constraints are added to the optimization problem to handle the separation of the different classes. SVM distinguishes by its theoretical and practical advantages, such as solid mathematical background, high generalization capability and ability to find global and non-linear classification solutions through using kernels [**93**], [**94**].

(3) Back Propagation Neural Network

A back propagation neural network employs one of the most popular neural network learning algorithms, the Back propagation algorithm. It has been used successfully for wide variety of applications. The back propagation algorithm performs learning on a multilayer feed-forward neural network. It iteratively learns a set of weights for prediction of the class labels. A multilayer feed-forward neural network consists of an input layer, one or more hidden layer, and an output layer. The inputs to the network correspond to the attributes measured for each training instance. These inputs pass through the input layer and are then weighted and fed simultaneously to second layer, known as a hidden layer. The outputs of the hidden layer units can be input to another hidden layer, and so on. The number of hidden layers is arbitrary, although in practice, usually only one is used. The weighted outputs of the last hidden layer are input to units making up the output layer [**95**] [**96**].

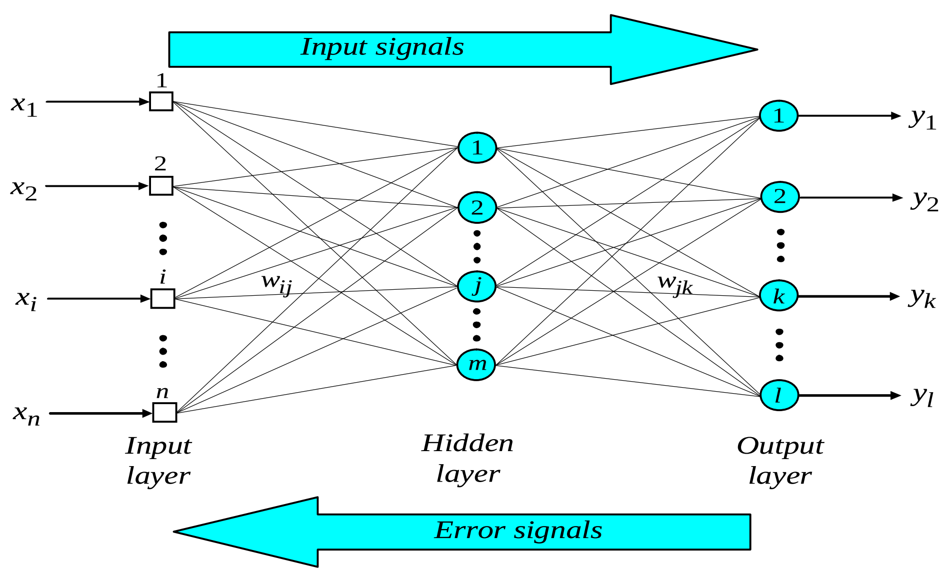

Figure 3.2 Three Layers Back Propagation Neural Networks

Figure 3.2, explain the two phases of BP: Forward pass phase and Backward pass phase. In Forward pass phase, computes ‘functional signal’, feed forward propagation of input pattern signals through network. In Backward pass phase, computes ‘error signal’, propagates the error backwards through network starting at output units (where the error is the difference between actual and desired output values) [**97**].

(4) Radial basis functions network

Radial basis functions network (RBF) is a type of neural network, which have a similar architecture to that of MLPs. RBF networks have become one of the most used classifier for regression, classification and function approximation applications. They are embedded into a two-layer feed-forward neural network, a hidden layer of radial kernels and an output layer of linear neurons. The hidden layer performs a non-linear transformation of input space. The output layer combines linearly the outputs of the hidden neurons to predict the desired targets as depicted in Figure 3.3. The RBF output is a value based on the radial distance of a feature vector from a center in the feature space. RBF centers are selected so as to match the distribution of training examples in the input feature space [**98**].

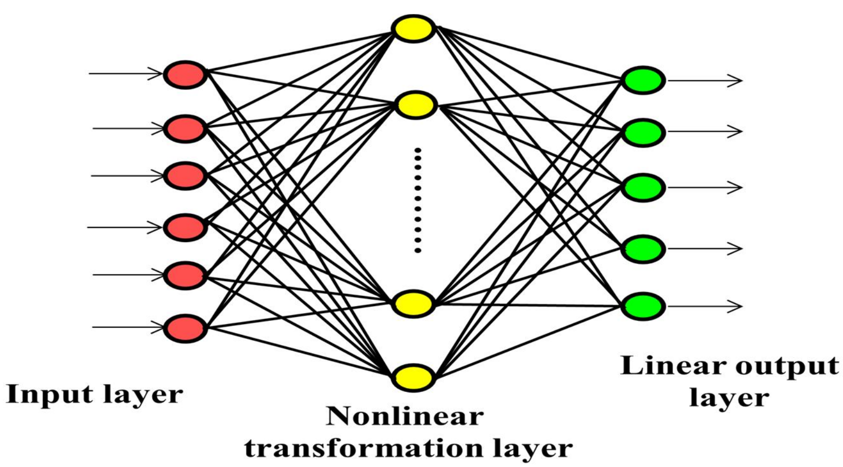

Figure 3.3 Radial basis function neural network

(5) C4.5

C4.5 is one of best-known and most widely-used learning algorithms proposed by Quinlan. It is An Improvement over ID3 algorithm by adding dealing with Missing data and numeric attributes, handling noisy data better and using Pre and post pruning to prevent overfitting. The decision tree generated from a set of training data by C4.5, using the concept of information entropy. At each node of the tree, C4.5 chooses one attribute of the data that most effectively splits. The attribute which has the highest information gain is chosen to make the decision tree [**99**], [**100**].

(6) Random forest

Random forest is one of the most accurate learning algorithms. It isan ensemble classifier that consists of many decision trees, where each tree is generated based on an independent set of random vectors of a data set. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification, the output is class that having the most votes over all the trees in the forest. RF runs efficiently on large databases and thousands of input variables. It gives estimates of what variables are important in the classification. And It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing. However, it have been observed to overfit for some datasets with noisy classification/regression tasks [**101**].

(7) Random Trees

Random Tree (RT) is a tree constructed from a set of trees that considers K randomly chosen attributes at each node. RT algorithm can deal with both classification and regression problems. All the trees are trained with the same parameters but on different training sets. These sets are generated from the original training set using the bootstrap procedure: for each training set, randomly select the same number of samples as in the original set. The samples are chosen with replacement. That is, some samples will occur more than once and some will be absent. At each node of each trained tree, not all the features are used to find the best split, but a subset of them randomly selected at each node expansion without any purity function check (such as information gain, gini index, etc.).The classification of the input feature vector works by classifying the input feature vector, it with every tree, and the output is class label that received the majority of votes. In case of a regression, the classifier response is the average of the responses over all the trees in the forest. As all trees, it has possibilities for explanation and visualization of its output. An accurate model can be generated by a combination of a set of random trees [**102**].

3.4 Meta learning ensembles methods

The main objective of ensemble is to try to improve the performance through using multiple classifiers and aggregate their predictions in decision making. In our experiments we use homogenous ensembles that use a single learning algorithm but manipulate training data to make it learn multiple models. We used the most widely used meta-learning algorithms Bagging and AdaBoost which have significant improvements in several classification problems. Section 3.3.1 explains the information of Bagging and section 3.3.2, explain the information of AdaBoost.

3.4.1 Bagging

Bagging was initially developed by Breiman. In a bagging ensemble, each individual classifier is trained using a different bootstrap of the data set. A bootstrap from a data set of N samples is a randomly drawn subset of N samples with replacement. The replacement allows samples to be drawn repeatedly. The final output is determined by simple vote among the component classifiers (See figure3.4) [**103**].

$Input: S: Training\ set; T: Number of\ iterations; n: Bootstrap\ size;$

$I: Weak\ learner$

$$Output: Bagged\ classifier: H(x) = sign\left(\sum_{t=1}^{T} h_t(x)\right)\ where$$

$h_t \in [1, -1]\ are\ the\ induced\ classifiers$

$for\ t = 1\ to\ T\ do$

$S_t \leftarrow Random\ Sample\ Replacement$

$h_t \leftarrow I(S_t)$

$endfor$

Figure 3.4 Bagging Algorithm

3.4.2 Boosting

Boosting (or Arcing , Adaptive reweighting and combining) [**104**], trains classifiers serially such that a new classifier should focus on those instances which were incorrectly classified in the last round and Combine the classifiers by letting them vote on the final prediction (like bagging). AdaBoost is specific boosting method. It adaptively weigh each data instance so that those are wrongly classified get high weight. Each boosting round learns a new (simple) classifier on the weighed dataset. Furthermore, those classifiers are weighed to combine them into a single powerful classifier whereas Classifiers that obtain low training error rate have high weight. In our experiments we use AdaBoost.M1 (see figure 3.5) which an extension of AdaBoost for multi classes [**105**].

$\boldsymbol{Input}: Training\ set\ S = \{x_i, y_i\}, i = 1,2,3, \dots, N; and\ y_i \epsilon C,$

$C = \{c_1, \dots, c_m\}; T: Number\ of\ iterations;\ I: Weaklearner$

$\boldsymbol{Output}: Boosted\ classifier:$

$$H(x) = arg\, max_{y \epsilon C} \sum_{t=1}^{T} ln\left(\frac{1}{B_t}\right)[h_t(x) = y]\ where\ h_t, B_t\ are\ the$$

$induced\ classifiers\ (with\ h_t(x) \epsilon C)\ and\ their\ assigned\ weights, respectively$

$D_1(i) \leftarrow 1/N\ for\ i = 1, \dots, N$

$\boldsymbol{for}\ t = 1\ to\ T\ \boldsymbol{do}$

$h_t \leftarrow I(S, D_i)$

$$\varepsilon_t \leftarrow \sum_{t=1}^{N} D_t(i)\,[h_t(x_i) \neq y]$$

$\boldsymbol{if} \varepsilon_t > 0.5\ \boldsymbol{then}$

$T \leftarrow t - 1$

$return$

$\boldsymbol{endif}$

$$B_t = \frac{\varepsilon_i}{1 - \varepsilon_i}$$

$D_{t+1}(i) = D_t(i). B_t^{1-[h_t(x_i) \neq y]} fori = 1, \dots, N$

$Normalize\ D_{t+1}\ to\ be\ a\ peoper\ distribution$

$\boldsymbol{endfor}$

Figure 3.5 AdaBoost.M1 Algorithm

3.5 Evaluation Metrics

Evaluation metrics is a critical issue in machine learning which used as indicator for the performance of machine learning algorithms. The standard evaluation metrics used are accuracy and error rate however, these metrics are not proper to handle imbalance classes as the overall accuracy be biased to the majority class regardless of the minority class with lower samples which leads to poor performance on it. In our experiments we use the most evaluation metrics related to imbalance classes which are recall (sensitivity) (1), specificity (2), precision (3), F-measure(4),(5). Recall and specificity are used to monitor the classification performance on each individual class. While precision is used in problems interested on highly performance on only one class, F-measure is used when the performance on both classes –majority and minority classes- needed to be high [**106**]. As an example for two class problem, metrics are derived from a confusion matrix which shown in Table 3.1.

Table 3.1: A 2x2 Confusion Matrix

Predicted Class

		+ve	**-ve**
Actual Class	**+ve**	True Positive (TP)	False Negative (FN)
Actual Class	**-ve**	False Positive (FP)	True Negative (TN)

$$Recall(true\ positive\ rate) = \frac{TP}{TP + FN} \qquad (1)$$

$$Specifity(true\ negative\ rate) = \frac{TN}{TN + FP} \qquad (2)$$

$$precision = \frac{TP}{TP + FP} \qquad (3)$$

$$F = \frac{(1 + \beta^2).(precision.Recall)}{\beta^2.precision + Recall} \qquad (4)$$

$where\ as\ \beta\ is$ non negative constant and generally is set to 1 so:

$$F = \frac{2.\,precision.\,Recall}{precision + Recall} \qquad (5)$$

3.6 Summary

In this chapter, we explained the experimental methodologies utilized in our experiments that include seven of the top machine learning and data mining algorithms, different imbalanced data sets for both two and multi class imbalance problems, meta learning methods and finally, performance measures used to evaluate our experiments.

CHAPTER FOUR

HANDLING TWO CLASS IMBALANCE PROBLEM

Two class imbalance problems occur when the instances of one class that forms the majority outnumber the instances of other minority class. The instances of the minority class are labeled as positive and those of the majority class as negative. In this Chapter we investigate two class imbalance problem through conducting several experiments and introduces a hybrid ensemble approach as a solution for two class problem. We begin with the design of all conducted experiments that contains four different phases, followed by results analysis and discussion.

4.1 Experiments Design Methodology

Our experiments have been carried out using seven classifiers (Naïve Bays (NB), Back Propagation Neural Network (BP), Support Vector Machine (SVM), Radial Basis Function Network (RBF), C4.5, Random Tree (RT), and Random Forest (RF)) (Section 3.3) with three different scenarios and phases and four two class imbalanced datasets (insurance fraud, German, Hepatitis and Haberman) with different imbalanced ratios (Section 3.1).

4.1.1 Phase One: Testing Classifiers Using the Original Data Distribution

In phase One, we tested the performance of the seven selected classifiers using the original distribution for different imbalanced data sets with different imbalanced ratios. The main objective of this phase is to compare the performance of different classifiers to reveal those sensitive to class imbalanced problem.

4.1.2 Phase Two: Using Resampling Methods

In phase two, we resampled data using undersampling by selecting random subsets from the majority classes that equal the size of minority classes. Then we tested the performance of seven classifiers using the new resampled data. The objective of this phase is to examine the effect of using undersampling on the performance of different classifiers.

Next, we resampled data using oversampling by increasing the minority class samples using SMOTE (Synthetic Minority Oversampling Technique). The new resampled data is used for classifiers training. The objective of this phase is to examine the effect of using oversampling on the performance of different classifiers when dealing with imbalanced classes.

4.1.3 Phase Three: Using Meta Learning Methods

In phase three, we repeated the previous experiments using meta-learning methods with homogenous ensemble methods such as Bagging and AdaBoost. The objective of this phase is to display the impact of the meta-learning method and their improvements on the performance of classifiers when dealing with imbalanced classes data using original data distribution (without applying any resampling method) and when applying resampling methods.

4.1.4 Phase Four: Using the proposed Approach

We have proposed a new method for dealing with two class imbalanced data sets. The basic idea for the proposed method is applying multiple resampling methods at various rates to construct several balanced datasets. Instead of designing multiple classifiers with the same dataset, we can manipulate the training set by resampling the original data using undersampling and oversampling.

Also, in our proposed method we integrate ensemble of multiple classifiers, which is one of the popular techniques being used recently to increase and boost the performance of weak learners.

Firstly we used SMOTE to oversample the minority class by adding new synthetic samples. SMOTE finds the k nearest neighbors for each minority sample according to the percentage of increase. Then, it selects randomly a point that lie on the line between each pair of nearest neighbors to generate the new added minority sample. In our method the percentage of increase is controlled by the number of repetition for the loop to generate different set in each repetition and overcome overfitting for the new model, which is a defection for oversampling.

In the next step, we applied undersampling by selecting randomly a subset from the majority class by size equal to the new oversampled minority class that obtained in the previous step. The objective of this step is to overcome the defection of undersampling, which causes loss of information by forming multiple subsets that contain different samples from majority class.

Then, the new trained dataset obtained by combining new balanced sets for minority and majority subsets. Then these multiple balanced datasets used to train the base classifiers of ensemble. The number of the base classifiers that formed for ensemble depends on the imbalance ratio between majority and minority class. After that, the trained ensemble model is evaluated on the testing data. The predicted class for any testing sample is calculated using average function for the output of all base classifiers with threshold value 0.5. Finally, the output is performance measures for the ensemble. Figure 4.1 explains the Algorithm of the proposed method.

```
Input: training data set (DT) and testing dataset (DS) and classifier C.
 Split training data into majority (M) and minority (N) class
 Calculate IR (imbalance ratio) = size M/size N
  For i=1 to ceil(IR*2)
     Ni= SMOTE(N,100/i);
     Mi=RUS(M,size(Ni));
     DTi= Ni U Mi;
     Build a component for the classifier C using DTi;
Evaluate the ensemble model using DS;
Find the predicted class using average function with threshold 0.5;
Output: TPR, TNR, precision and F-measure
```

Figure 4.1 Algorithm of the proposed method for handling two class problem

4.2 Results Analysis and Discussion

In our experiments we used four datasets with different imbalance ratios as summarized in Table 4.1. Using seven selected classifiers NB, SVM, BP, RBF, C4.5, RF and RTs.

Table 4.1. Datasets summary

Data set	#of instances	# of Attributes	# of instances in Majority class	# of instances in Minority class	Imbalance Ratio
Insurance fraud	15420	32	14497 (94%)	923 (6%)	15.7
Hepatitis	155	20	123 (79%)	32 (21%)	3.8
German	1000	21	700 (70%)	300 (30%)	2.3
Haberman	306	4	225 (74%)	81 (26%)	2.8

4.2.1 Results Analysis and Discussion for Phase One

Table 4.2 depict results in term of accuracy for all data sets and Tables 4.3-4.6 explain the results for each data sets using TN rate, TP rate, precision and F- measure as evaluation measures.

Table 4.2. Performance of classifiers on different datasets in term of accuracy

Method	Insurance Fraud	German	Hepatitis	Haberman
NB	0.93	0.75	0.83	0.75
SVM	0.94	0.7	0.79	0.74
BP	0.94	0.73	0.82	0.73
RBF	0.94	0.74	0.83	0.74
C4.5	0.94	0.71	0.81	0.72
RF	0.97	0.74	0.84	0.69
RT	0.96	0.66	0.77	0.64

Table 4.3 Performance of different classifiers on Insurance Fraud data set using the original data distribution

Method	TNR	TPR	Prec	F-M
NB	0.98	0.11	0.27	0.16
SVM	1.00	0.00	0 .00	0.00
BP	1.00	0.09	0.86	0.17
RBF	1.00	0.01	0.00	0.01
C4.5	1.00	0.04	0.83	0.07
RF	1.00	0.57	0.97	0.72
RT	0.98	0.66	0.70	0.68

Table 4.4. Performance of different classifiers on German data set using the original data distribution

Method	TNR	TPR	Prec	F-M
NB	.86	.50	.8	.83
SVM	1.00	0.00	0.00	0.00
BP	.82	.52	.55	.53
RBF	.87	.45	.60	.51
C4.5	.84	.39	.52	.44
RF	.88	.41	.59	.49
RT	.75	.45	.43	.44

Table 4.5. Performance of different classifiers on Hepatitis data set using the original data distribution

Classifier	TNR	TPR	Prec	F-M
NB	.89	.63	.61	.62
SVM	1.00	0.00	0.00	0.00
BP	.87	.63	.90	.88
RBF	.88	.66	.58	.62
C4.5	.90	.44	.54	.48
RF	.89	.63	.61	.62
RT	.85	.50	.46	.48

Table 4.6. Performance of different classifiers on Haberman data set using the original data distribution

Method	TNR	TPR	Prec	F-M
NB	.94	.21	.58	.31
SVM	.99	.03	.5	.05
BP	.88	.30	.48	.37
RBF	.95	.17	.54	.26
C4.5	.87	.30	.45	.36
RF	.84	.26	.37	.30
RT	.77	.30	.32	.31

Compared to the obtained results for insurance fraud data set, as evident from Table 4.2 we find that all classifiers have overall accuracy up to (92%). But if we compare the performance using detection rate for each class: TP rate and TN rate as depicted in Table 4.3, we find out that the detection rates for majority class (True negative rates) are always up to (98%) regardless of the classifier used. Contrary to these results, the highest detection rates for the minority class (TP rates) are 65%, 56%, which are obtained by random tree and random forests respectively but all other classifiers have given TP rates less than 1%.

The same thing occurs with other three datasets, which are depicted clearly in Figures 4.2- 4.5. We note that the higher detection rates are for the negative class and the lower detection rates are for the positive class which emphasize that the overall accuracy are biased towards negative class. Obviously, we can also deduce that all the used classifiers are very sensitive for imbalanced classes but the most influenced one is SVM which is biased totally to the negative class and produced TP rates equal to zero. The lower TP rates also produced lower precision and lower F- measure.

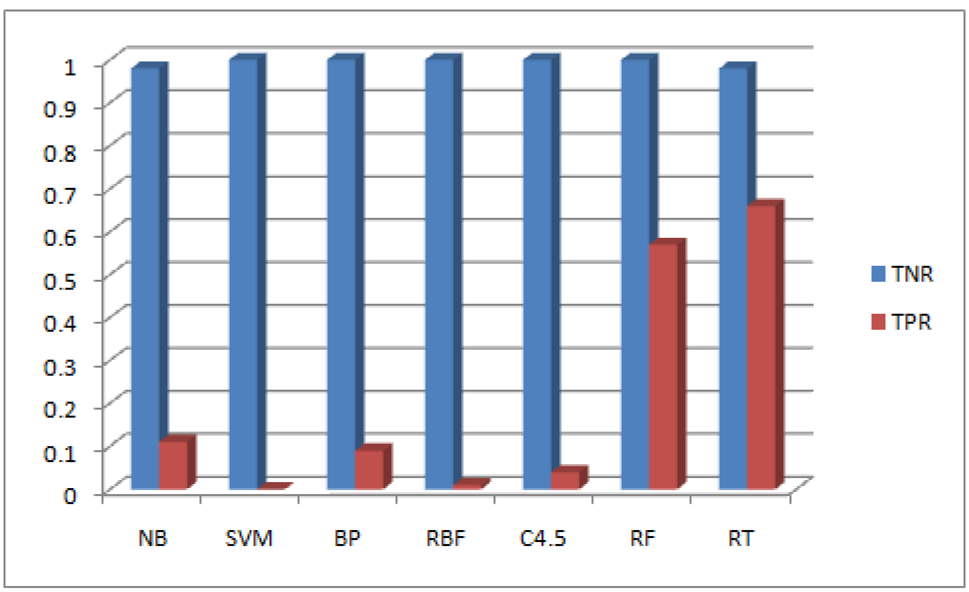

Figure 4.2 Detection rates for positive and negative classes in Insurance fraud data set

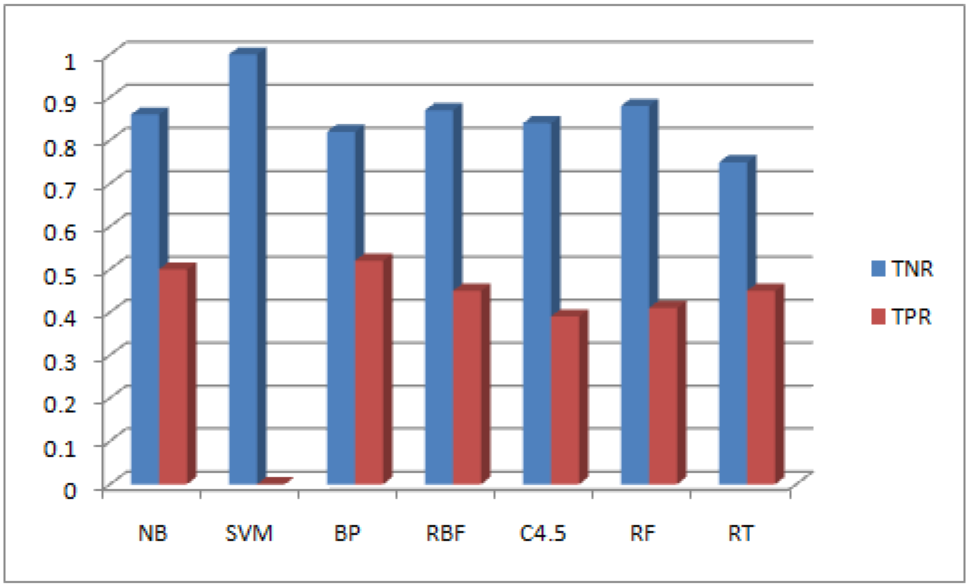

Figure 4.3 Detection rates for positive and negative classes in German data set

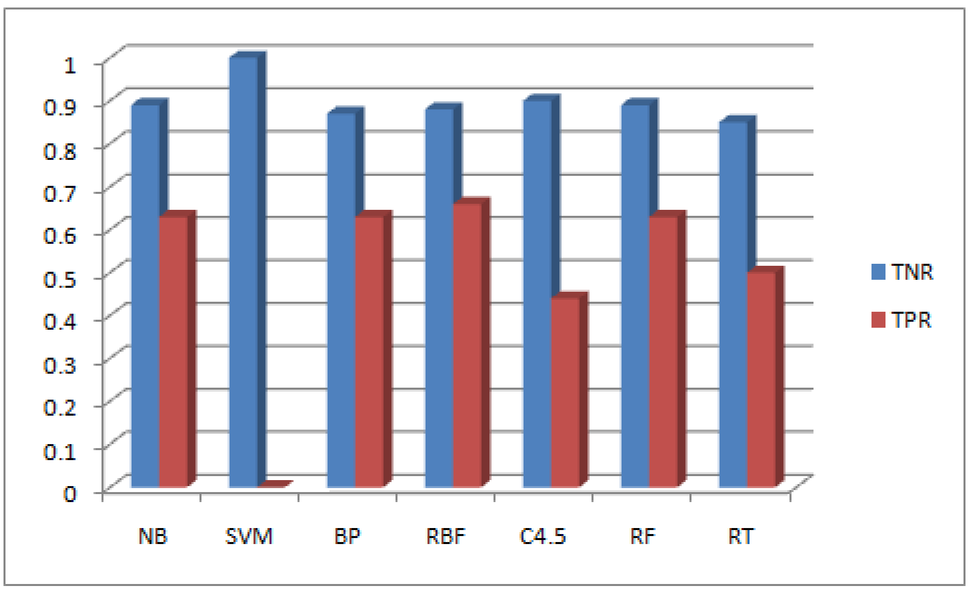

Figure 4.4 Detection rates for positive and negative classes in Hepatitis data set

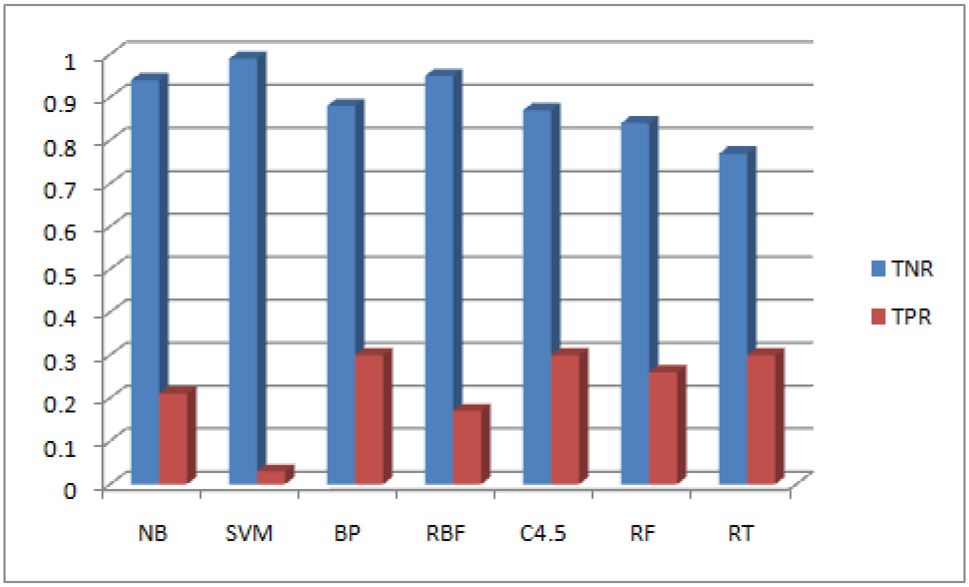

Figure 4.5 Detection rates for positive and negative classes in Haperman data set

4.2.2 Results Analysis and Discussion for Phase Two

In the next phase, we have balanced data using undersampling by selecting randomly a subset from the majority (negative) class that equal to the size of minority (positive) class. The results after using undersampling depicted in Tables 4.7-4.10.

Table 4.7. Performance of classifiers on Insurance Fraud data set using undersampling

Method	TNR	TPR	Prec	F-M
NB	0.59	0.94	0.13	0.23
SVM	0.61	0.90	0.13	0.23
BP	0.61	0.90	0.13	0.23
RBF	0.63	0.85	0.13	0.23
C4.5	0.59	0.96	0.13	0.24
RF	0.66	0.91	0.15	0.26
RT	0.63	0.90	0.14	0.24

Table 4.8. Performance of different classifiers on German data set using undersampling

Method	TNR	TPR	Prec	F-M
NB	.88	.71	.86	.77
SVM	.16	.87	.51	.64
BP	.79	.74	.78	.76
RBF	.81	.75	.8	.77
C4.5	.81	.73	.80	.76
RF	.78	.76	.78	.77
RT	.71	.71	.71	.71

Table 4.9. Performance of classifiers on Hepatitis data set using undersampling

Method	TNR	TPR	Prec	F-M
NB	.91	.88	.90	.89
SVM	.34	.5	.43	.46
BP	.81	.91	.83	.87
RBF	.84	.94	.86	.90
C4.5	.88	.84	.87	.86
RF	.81	.94	.83	.88
RT	.78	.81	.79	.8

Table 4.10. Performance of classifiers on Haberman data set using undersampling

Method	TNR	TPR	Prec	F-M
NB	.93	.75	.91	.82
SVM	.60	.82	.67	.73
BP	.92	.72	.89	.80
RBF	.84	.75	.82	.79
C4.5	.90	.67	.87	.76
RF	.84	.77	.83	.80
RT	.74	.78	.75	.76

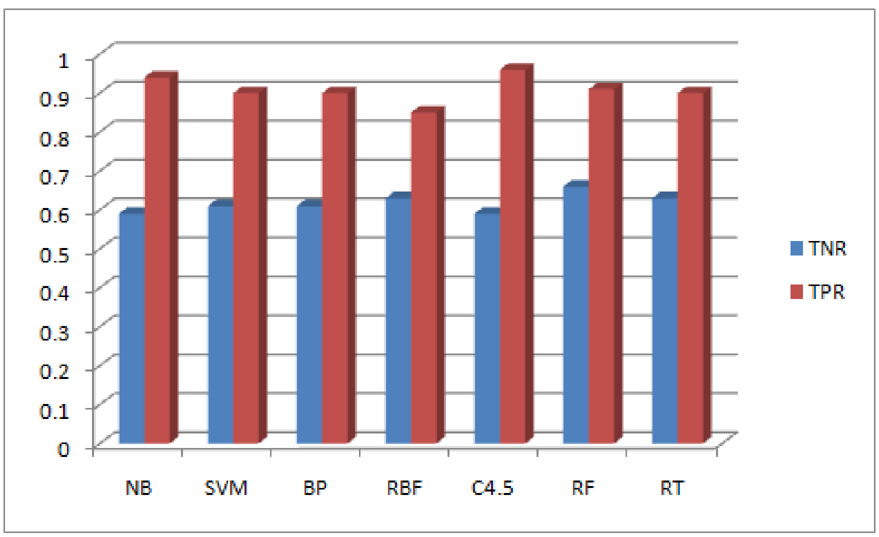

Figure 4.6. Detection rates of negative and positive classes insurance fraud data set when using undersampling

From Tables 4.7-4.10, we noticed that there are clear improvements in the true positive rates, however; the TNRs have become less and the other performance measures such as precision and f-measure have degraded. So these results have reflected the defection of undersampling which resulted in loss of information by removing significant samples from the negative class. This case is clearly depicted in Figure 4.6 which explain the results of insurance fraud data set (data set with highest imbalance ratio) when using undersampling. However, in data sets with low imbalanced ratio there is a proportional degradation in TN rates because the number of excluded samples was low in comparison with loss occurred in high imbalanced data sets.

In the third phase, we have oversampled data by applying SMOTE. SMOTE used to generate new synthetic samples and added them to minority class. Tables 4.11-4.14 explain results when using oversampling. Obviously, these results explain that SMOTE has significantly improved the performance of all classifiers. However, in insurance fraud data set we realize that there is a clear degradation on the detection rates for the negative class and there are proportional improvements on the true positive rates as depicted in Figure. 4.7.

Table 4.11. Performance of classifiers on Insurance Fraud data set using oversampling

Method	TNR	TPR	Prec	F-M
NB	0.72	0.67	0.14	0.23
SVM	0.86	0.51	0.19	0.28
BP	0.95	0.40	0.33	0.36
RBF	0.79	0.57	0.16	0.25
C4.5	0.95	0.42	0.37	0.40
RF	0.99	0.66	0.79	0.72
RT	0.96	0.73	0.56	0.63

Table 4.12. Performance of classifiers on German data set using oversampling

Method	TNR	TPR	Prec	F-M
NB	.79	.80	.79	.79
SVM	.91	.35	.80	.49
BP	.79	.79	.79	.79
RBF	.79	.76	.78	.77
C4.5	.78	.77	.78	.77
RF	.82	.80	.81	.81
RT	.71	.77	.73	.75

Table 4.13. Performance of classifiers on Hepatitis data set using oversampling

Method	TNR	TPR	Prec	F-M
NB	.87	.84	.87	.86
SVM	.95	.51	.92	.65
BP	.81	.95	.84	.89
RBF	.82	.90	.84	.87
C4.5	.83	.87	.84	.85
RF	.83	.95	.85	.90
RT	.85	.83	.86	.84

Table 4.14. Performance of classifiers on Haberman data set using oversampling

Method	TNR	TPR	Prec	F-M
NB	.88	.36	.76	.49
SVM	.72	.76	.73	.74
BP	.74	.58	.69	.63
RBF	.71	.64	.68	.66
C4.5	.70	.71	.70	.70
RF	.75	.70	.74	.72
RT	.73	.72	.23	.72

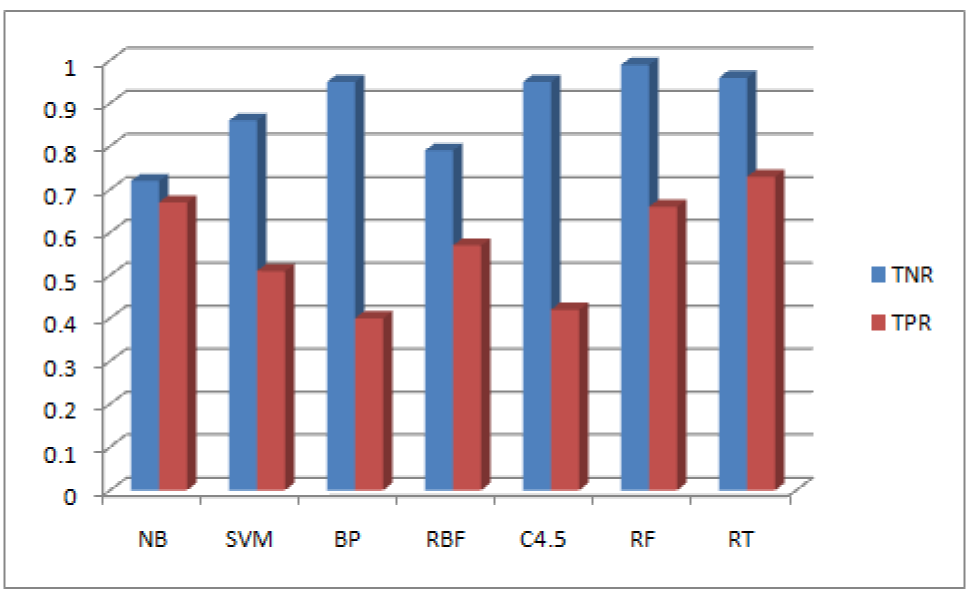

Figure 4.7. Detection rates of negative and positive of insurance fraud data set when using oversampling

4.2.3 Results Analysis and Discussion for Phase Three

Further experiments repeated using Bagging and AdaBoost meta learning methods and the same tested classifiers as bases classifiers. Tables 4.15 - 4.18 explain results when applying Bagging and AdaBoost using the imbalanced original data distribution and using resampling balanced data. Clearly we note that there are no significant effects on the performance using original data distribution, but the performance got better after applying meta learning methods on balanced data using undersampling or oversampling resampling methods.

Table 4.15. Performance of classifiers when using meta Learning methods on Insurance Fraud data set

Method		Original Data Distribution				Balancing Data Using Undersampling				Balancing Data Using Oversampling			
		TNR	**TPR**	**Prec**	**F-M**	**TNR**	**TPR**	**Prec**	**F-M**	**TNR**	**TPR**	**Prec**	**F-M**
Using Bagging	NB	.98	.09	.25	.13	.57	.93	.13	.23	.72	.67	.14	.23
	SVM	1	.01	1	.01	.60	.90	.13	.23	.86	.51	.19	.29
	BP	1	.16	.97	.28	.79	.91	.16	.28	.95	.46	.40	.43
	RBF	.86	.51	.19	.28	.66	.81	.14	.23	.79	.58	.16	.25
	C4.5	1	.04	.83	.07	.64	.93	.15	.25	.99	.66	.79	.72
	RF	1	.34	1	.51	.65	.94	.15	.26	.99	.62	.78	.69
	RT	1	.57	.97	.72	.66	.91	.15	.26	.99	.66	.79	.72
Using AdaBoost	NB	.98	.11	.25	.15	.65	.84	.14	.24	.72	.67	.14	.23
	SVM	.99	.39	.67	.49	.67	.82	.14	.24	.93	.48	.30	.37
	BP	1.0	.09	.86	.17	.65	.90	.15	.25	.95	.40	.33	.36
	RBF	.99	.06	.29	.09	.70	.77	.14	.24	.83	.52	.17	.26
	C4.5	.99	.67	.88	.76	.70	.91	.17	.29	.99	.67	.74	.70
	RF	1.0	.64	.92	.75	.69	.90	.16	.27	.99	.66	.79	.72
	RT	.98	.67	.74	.71	.6	.90	.13	.23	.98	.68	.67	.68

Table 4.16. Performance of classifiers when using meta Learning methods on German data sets

Method		Original Data Distribution				Balancing Data Using Undersampling				Balancing Data Using Oversampling			
		TNR	**TPR**	**Prec**	**F-M**	**TNR**	**TPR**	**Prec**	**F-M**	**TNR**	**TPR**	**Prec**	**F-M**
Using Bagging	NB	.87	.51	.13	.56	.88	.72	.85	.78	.79	.79	.79	.79
	SVM	1.00	0	0	0	.62	.38	.5	.43	.92	.33	.80	.47
	BP	.87	.53	.64	.58	.82	.76	.81	.78	.81	.81	.81	.81
	RBF	.89	.43	.63	.51	.82	.77	.81	.79	.78	.79	.78	.79
	C4.5	.87	.43	.58	.49	.87	.71	.85	.77	.80	.79	.80	.80
	RF	.92	.36	.67	.47	.84	.76	.87	.79	.83	.86	.83	.83
	RT	.88	.41	.59	.49	.78	.76	.78	.77	.82	.80	.81	.81
Using AdaBoost	NB	.87	.49	.62	.55	.82	.74	.81	.77	.78	.81	.79	.80
	SVM	1.00	.01	.4	.01	.50	.49	.50	.49	.92	.34	.80	.47
	BP	.81	.53	.55	.54	.79	.74	.78	.76	.79	.78	.79	.79
	RBF	.85	.47	.58	.52	.78	.76	.77	.77	.76	.78	.76	.77
	C4.5	.79	.46	.48	.47	.75	.75	.75	.75	.81	.78	.80	.79
	RF	.89	.42	.62	.50	.83	.74	.81	.78	.84	.81	.83	.82
	RT	.79	.45	.48	.46	.71	.7	.71	.70	.74	.74	.74	.74

Table 4.17. Performance of classifiers when using meta Learning methods on Hepatitis data sets

Method		Original Data Distribution				Balancing Data Using Undersampling				Balancing Data Using Oversampling			
		TNR	**TPR**	**Prec**	**F-M**	**TNR**	**TPR**	**Prec**	**F-M**	**TNR**	**TPR**	**Prec**	**F-M**
Using Bagging	NB	.89	.66	.62	.64	.88	.94	.88	.91	.86	.86	.86	.86
	SVM	1.00	0	0	0	.23	.78	.5	.61	.85	.53	.79	.64
	BP	.033	1.0	1.0	.06	.23	1.0	.56	.72	.06	.99	.52	.69
	RBF	.94	.69	.73	.71	.91	.88	.90	.89	.86	.90	.87	.89
	C4.5	.93	.56	.67	.61	.88	.81	.87	.84	.81	.92	.84	.88
	RF	.94	.5	.67	.57	.88	.88	.88	.88	.87	.92	.88	.90
	RT	.894	.63	.61	.62	.81	.94	.83	.88	.83	.95	.85	.90
Using AdaBoost	NB	.94	.50	.68	.56	.84	.84	.84	.84	.89	.88	.89	.88
	SVM	.99	0	0	0	.47	.56	.51	.54	.96	.42	.92	.58
	BP	.90	.53	.59	.56	.81	.91	.83	.87	.86	.95	.87	.91
	RBF	.94	.63	.74	.68	.84	.88	.85	.86	.85	.92	.87	.89
	C4.5	.94	.53	.68	.60	.78	.81	.79	.8	.89	.93	.90	.91
	RF	.90	.47	.56	.51	.75	.91	.78	.84	.85	.94	.86	.90
	RT	.81	.44	.37	.4	.94	.75	.92	.83	.84	.87	.85	.86

Table 4.18. Performance of classifiers when using meta Learning methods on Haberman data sets

Method		Original Data Distribution				Balancing Data Using Undersampling				Balancing Data Using Oversampling			
		TNR	**TPR**	**Prec**	**F-M**	**TNR**	**TPR**	**Prec**	**F-M**	**TNR**	**TPR**	**Prec**	**F-M**
Using Bagging	NB	.94	.20	.55	.29	.92	.75	.90	.82	.89	.36	.76	.50
	SVM	.98	.06	.5	.11	.59	.82	.66	.73	.70	.74	.71	.71
	BP	1.0	0	0	0	.57	.51	.54	.52	.13	.92	.51	.66
	RBF	.94	.19	.54	.28	.82	.77	.81	.79	.74	.63	.71	.67
	C4.5	.88	.22	.4	.29	.92	.72	.89	.80	.70	.78	.72	.75
	RF	.84	.25	.35	.29	.83	.75	.81	.78	.72	.81	.74	.77
	RT	.84	.26	.37	.30	.84	.77	.83	.80	.75	.70	.74	.72
Using AdaBoost	NB	.91	.28	.52	.37	.93	.75	.91	.82	.76	.55	.69	.61
	SVM	.87	.15	.29	.20	.52	.83	.63	.77	.70	.78	.72	.75
	BP	.88	.28	.47	.35	.85	.73	.83	.79	.78	.60	.73	.66
	RBF	.94	.21	.55	.30	.76	.74	.75	.75	.72	.64	.69	.66
	C4.5	.84	.42	.48	.45	.92	.72	.89	.80	.71	.74	.71	.73
	RF	.82	.33	.40	.36	.72	.75	.73	.74	.75	.71	.73	.72
	RT	.76	.26	.28	.27	.76	.77	.76	.76	.75	.73	.74	.73

From all previous results we can report that all classifiers get a significant improvement when applying undersampling or oversampling with/without meta learning methods. However, undersampling and oversampling improve the performance on the positive class and inversely degrade the performance on the negative class. This is due to loss of significant and important samples excluded by undersampling and overfitting caused by oversampling. So, this affirms that using undersampling and oversampling until a full balance may not be an optimal solution and the defection of them are increased whenever a high imbalanced ratio founded.

4.2.4 Results Analysis and Discussion for Phase Four

Further experiments are suggested to use different resampling rates. However, it is not possible to know a given domain prefer undersampling or oversampling. So we decided to create a combination approach and propose a hybrid ensemble that considers both resampling methods at different rates (see section 4.1.1). Table 4.19 explains the performance results for the proposed method when applied in different datasets in term of TNR, TPR, precision and f-measure. The results shown in Table 4.19 indicate that our proposed method the proposed method yielded good results comparing to the other tested methods in term of True negative rates.

In term of True positive rates, when being compared with the basic classifiers, Bagging, AdaBoost or SMOTE with/out Bagging or AdaBoost, we note that our proposed method achieved superior performance results. Form the other side, our proposed method mostly gives TPRs closely to those performed by using undersampling with/out Bagging/AdaBoost. However, our proposed method give higher TPRs without decreases the detection rates for the negative class or causing bias to the one class due to overfitting.

In term of precision, our proposed method achieved competitive results on most data sets. We note that using Bagging with SVM and BP in insurance fraud data set and using BP with bagging in hepatitis data set resulted in a higher precision than our method but this occur due to the total biased for these methods to the positive class which is tackled by the proposed method.

In term of f-measure, our proposed method performed a significant improvements and superior results in comparison to the other tested methods.

Finally, we conclude that our proposed method performed well in all data sets with different imbalance ratios and significantly outperformed other methods in insurance fraud, a data set with highly imbalance ratio.

Table 4.19. Performance of the proposed method on different data sets

Data set	Method	TNR	TPR	Prec	F-M
Insurance fraud	NB	.870	.840	.870	.860
	SVM	.985	.866	.799	.831
	BP	.820	.770	.810	.790
	RBF	.918	.889	.909	.899
	C4.5	1.00	1.00	.998	.999
	RF	1.00	1.00	.998	.999
	RT	1.00	1.00	.998	.999
German	NB	.793	.774	.793	.793
	SVM	.891	.895	.882	.889
	BP	.891	.895	.882	.889
	RBF	.918	.889	.909	.899
	C4.5	.933	.896	.924	.91
	RF	.924	.881	.914	.897
	RT	.863	.858	.852	.855
Hepatitis	NB	.919	.875	.913	.894
	SVM	.751	.725	.742	.733
	BP	.889	.948	.892	.919
	RBF	.899	.917	.898	.907
	C4.5	.869	.927	.873	.899
	RF	.909	.948	.91	.929
	RT	.909	.948	.91	.929
Haberman	NB	.927	.753	.91	.824
	SVM	.598	.815	.667	.733
	BP	.915	.716	.892	.795
	RBF	.841	.753	.824	.787
	C4.5	.902	.667	.871	.755
	RF	.841	.765	.827	.795
	RT	.744	.778	.75	.764

CHAPTER FIVE

HANDLING MULTI CLASS IMBALANCED PROBLEM

The multi class imbalance problem is an extension of the traditional two class imbalanced data where a data set consists of k classes instead of two. While imbalance is said to exist in the binary class imbalance problem when one class severely outnumber the other class, extended to multiple classes the effects of imbalance are even more problematic. In this chapter we investigated multi class imbalance problem through conducting several experiments and introduced a hybrid ensemble approach as a solution for multi class problem. We introduce the experiments that contains three different phases including direct multi class classification, homogenous ensembles using meta learning methods and finally the proposed hybrid ensemble for multi class classification. We also present the results of the analysis using different evaluation measures.

5.1 Experiments Design

The phases followed in performing the targeted experiments are outlined in the following sub sections:

5.1.1 Phase One: Testing Classifiers Using the Original Data Distribution

Firstly, in our experiments we implemented multi class classification by directly adapted specific algorithms using single classifier with multiple outputs. We tested the performance of seven selected classifiers (Naïve Bays (NB), Support Vector Machine (SVM), Back Propagation Neural Networks (BP), Radial Basis Function Network (RBF), C4.5, Random Tree (RT), and Random Forest (RF)). The main objective of this phase is to compare the performance of different classifiers to reveal those sensitive to class imbalanced class problem. However, as we were dealing with imbalance class problem, the overall accuracy was biased to the majority class regardless the minority classes with lower samples, which leads to poor performance on the minority classes.

5.1.2 Phase Two: Using Meta Learning Methods

In this phase we compare the classifiers' performance when applied them as base classifiers for homogenous ensemble methods such as Bagging and AdaBoost. From this phase we display the impact of using homogenous ensemble approaches and their effects on the performance of classifiers when dealing with imbalanced classes data.

5.1.3 Phase Three: The proposed Approach Methodology

Our proposed solution is based into multi binary classification, which is accomplished by transformation and decomposition of the multi class problem into small binary sub problems. Then ensemble is made of those binary classifiers and the final output of the ensemble is a combination of binary classifiers outputs. There are several approaches for multi binary classification differ on the used decomposition and integration techniques. The most popular ones are One against All, One against One and Error Correcting Outputs code.

Our proposed method based on Error Correcting Output Code (ECOC) multi binary classification approach was proposed by Ditterich and Bakiri [**107**]. This approach converts the k multi class problem into N binary sub problems. Instead of assigned a class label, it assigns a binary string of length N which referred as a codeword. The code words represented by a KxN code matrix such that each row is associated with specific class and each column is associated with specific classifier output [**108**]. When constructing the code matrix, row and column separation must be considered. The best row separation is determined by well separating distance measure from each other rows. The best column separation is determined by uncorrelated output bit forming each classifier i.e. each bit function should be uncorrelated with function for other bit positions (columns must neither identical nor complementary).

The output code word for each class can be determined using two methods [**107**]:

a. One per Class coding: Specify classifier for each class, the classifier's output should be 1 for this class.

Table 5.1. One per Class coding

Class 1	1000
Class 2	0100
Class 3	0010
Class 4	0001

b. Distributed Output Coding: each class assigned a unique code word from 0 to

2^N-1 where N the number of classifiers.

Table 5.2. Distributed output coding

Class 1	00000
Class 2	00111
Class 3	11001
Class 4	11110

Our ECOC solution consists of two stages: encoding and decoding. The encoding stage includes the design of code matrix according to one per class coding methods. Then each classifier is trained on a two meta class problem. In decoding stage, run all classifiers. Aggregate their outputs to obtain code word. The obtained codeword compared with all code words in the code matrix using specific distance measure to determine the closest code word as a final output for ECOC ensemble. Figure 5.1 the Pseudo code of the algorithm of the proposed method.

Training Phase

1. Given a problem with m classes, create an m x m binary matrix M according to one per class coding method.

2. Each class is assigned one row of M (Each column divides the entire class space into two parts).

3. Train the base classifier to learn the n binary functions (one for each column since each column divides the data set into two groups).

Test Phase

1. Apply each of the n single-bit classifiers to the test sample.

2. Combine the predictions to form a binary string of length n (code word).

3. Classify to the class with the nearest code word using weighted hamming distance where as:

The hamming distance between two vectors u; v:

d(u; v) = the number of places where u and v differ.

Weight = 1/ number of the instances for the specific class.

Figure 5.1 The Pseudo code of the proposed method for multi class problem

The training phase of the proposed method starts with constructing an m x m code words matrix according to one per class method. Each class is assigned one row that has one position set for specific class and zero in positions for the other classes. Then each classifier is trained on specific Meta binary problem. In the testing phase, we apply each of the n single-bit classifiers to the test sample and Combine the predictions to form a binary string of length n (code word). The class is determined by finding the nearest code word using weighted hamming distance. We adapt ECOC for class imbalance problem by adding a weight to the distance function that is equal to 1/ no of instances in the class. This weight is significant and important for decision when there is an instance has same distance between more than one classes. Clearly it gives the higher priority to the minority classes.

5.2 Results Analysis and Discussion

In our experiments we used five data sets with different imbalance ratios as summarized in Table 5.3 (Section 3.1.2) and seven selected classifiers NB, SVM, BP, RBF, C4.5, RF and RTs (Section 3.3).

Table 5.3. Dataset summary

Data set	Total number of instance	# of Attributes	# of classes	# of instances in Majority class	# of instances in Minority class	# of instances per class	Imbalance Ratio
Intrusion Detection	5092	42	5	3002	27	1000: 500: 3002: 27: 563	111.2
Thyroid	3772	22	3	3488 (92%)	93 (2%)	93: 191: 3488	37.5
Lymphography	148	19	4	81 (55%)	2 (1%)	2: 81: 61: 4	55
Glass	214	10	6	76 (36%)	9 (4%)	70: 76: 17: 13: 9: 29	8.4
Landsat	4435	37	6	1072 (24%)	415 (9%)	1072: 479: 961: 415: 470: 1038	2.6

5.2.1 Results Analysis and Discussion for Phase One

Table 5.4 depicts results in term of accuracy for all data sets and Tables 5.5-5.9 explain the results for each data sets using detection rate for each class as evaluation measures.

Table 5.4. Performance of classifiers on different datasets in term of accuracy

Method	Intrusion Detection	Thyroid	Lymphography	Glass	Landsat
NB	0.834	0.957	0.811	0.495	0.835
SVM	0.874	0.927	0.777	0.692	0.038
BP	0.993	0.962	0.824	0.673	0.887
RBF	0.942	0.936	0.811	0.659	0.831
C4.5	0.948	0.997	0.784	0.659	0.847
RF	0.988	0.995	0.824	0.738	0.847
RT	0.828	0.985	0.791	0.696	0.835

Table 5.5. Detection rates per class in Intrusion Detection data set

Method	Class 1	Class 2	Class 3	Class 4	Class 5
NB	0.99	0.57	0.89	0.44	0.3
SVM	0.63	0.57	0.996	0	0.98
BP	0.997	0.994	0.997	0.56	0.975
RBF	0.99	0.56	0.99	0.44	0.94
C4.5	0.995	0.85	0.99	0.48	0.67
RF	0.997	0.99	0.99	0.48	0.95
RT	0.58	0.85	0.95	0.16	0.54

Table 5.6. Detection rates per class in Thyroid data set

Method	class 1	class 2	class 3
NB	0.828	0.366	0.993
SVM	0	0	1
BP	0.795	0.497	0.992
RBF	0.616	0	0.996
C4.5	0.978	1	0.997
RF	0.989	0.984	0.996
RT	0.903	0.864	0.994

From the obtained results for Intrusion Detection data set and as evident from Tables 5.4 and 5.5, we noted that all classifiers have overall accuracy (up to 82%) but if we compare this with detection rates for each class we find that all classifiers have highest detection rates for class 3 and class 1 (the most majority classes) except SVM, which has lowest detection rate for class 1. Also, All classifiers have rather good detection rates for class 2 and class5 (minority classes but with large numbers of samples) however, we find that all classifiers have bad detection rates for class 4 which is a class with lowest number of samples and SVM is the most effective classifier which has detection rate equal 0 for class 4. So as we are dealing with imbalance class problem the overall accuracy biased to the majority classes regardless of the minority class with lower samples, which leads to poor performance on the minority class.

As evident from Table5.6, we note that all classifiers have highest detection rates for class 3 (majority class). C4.5, RF, RT are the most robust to classify other classes correctly however, NB, SVM, BP, RBF have lower detection rates for class1 and class 2 (minority classes) and SVM has the lowest detection rates equal 0for them.

Table5.7. Detection rates per class in Landsat data set

Method	Class 1	Class 2	Class 3	Class 4	Class 5	Class 7
NB	0.968	0.885	0.933	0.612	0.838	0.81
SVM	1	0	0	0	0	0.033
BP	1	0.957	0.948	0.633	0.81	0.92
RBF	0.935	0.967	0.981	0.327	0.857	0.83
C4.5	0.935	0.967	0.926	0.456	0.867	0.866
RF	0.935	0.967	0.926	0.456	0.867	0.866
RT	1	0.957	0.862	0.503	0.867	0.857

Table5.7 depicts the detection rates per class in Landsat data set. We notice that all classifiers have lowest detection rates for class 4 (the class with minimum number of instances). Clearly, SVM biased totally to the most majority class (class 1) and failed to detect other classes.

Table 5.8. Detection rates per class in Lymphography data set

Method	Class 1	Class 2	Class 3	Class 4
NB	0	0.877	0.754	0.75
SVM	0	0.914	0.672	0
BP	0	0.864	0.787	1
RBF	0	0.889	0.77	0.25
C4.5	0	0.852	0.705	1
RF	0	0.938	0.721	0.5
RT	0.5	0.827	0.77	0.5

Table 5.8 explains the detection rates per class in Lymphography data set. It depicts that all classifier have fail to detect class 1 (the class with lowest number of instances) except RT. Also, for class 4only NB, BP and C4.5 succeeded to classify it.

Table 5.9. The detection rates per class in Glass data set

Method	class 1	class 2	class 3	class 5	class 6	class 7
NB	0.714	0.197	0.353	0.231	0.889	0.828
SVM	0.814	0.75	0	0.692	0.222	0.793
BP	0.8	0.658	0	0.538	0.667	0.862
RBF	0.729	0.632	0.118	0.538	0.778	0.897
C4.5	0.714	0.566	0.294	0.846	0.889	0.828
RF	0.814	0.737	0.353	0.692	1	0.724
RT	0.7	0.75	0.353	0.538	0.667	0.828

Table5.9 explains the detection rates per class in Glass data set. It depicts that all classifiers have lowest detection rates for class 3 and class 5. SVM and BP are the most affected ones to detect class 3.

5.2.2 Results Analysis and Discussion for Phase Two

In the next phase, we used homogenous ensembles Bagging and AdaBoost and the same classifiers as base classifiers. Tables 5.10- 5.14 depict results when using Bagging and Tables 5.15- 5.19 depict results when using AdaBoost.

Table 5.10. Detection rates when using Bagging in Intrusion Detection data set

Method	class 1	class 2	class 3	class 4	class 5
NB	0.99	0.58	0.72	0.44	0.3
SVM	0.64	0.57	0.998	0.36	0.98
BP	0	0	0.717	0	0.011
RBF	0.995	0.91	0.71	0.4	0.85
C4.5	0.995	0.85	0.987	0.48	0.68
RF	0.997	0.996	0.998	0.4	0.98
RT	0.997	0.99	0.99	0.48	0.95

Table 5.11. Detection rates when using Bagging in Thyroid data set

Method	class 1	class 2	class 3
Bagging+NB	0.808	0.322	0.989
Bagging+SVM	0	0	1
Bagging+BP	0.041	0	1
Bagging+RBF	0.603	0	0.995
Bagging+C4.5	0.973	1	0.993
Bagging+RF	0.986	1	0.992
Bagging+RT	1	1	0.992

Table 5.12. Detection rates when using Bagging in Landsat data set

Method	Class 1	Class 2	Class 3	Class 4	Class 5	Class 7
Bagging+SVM	1	0	0	0	0	0.031
Bagging+BP	1	0	0	0	0	0
Bagging+RBF	0.938	0.971	0.964	0.364	0.777	0.843
Bagging+C4.5	0.935	0.967	0.967	0.517	0.867	0.917
Bagging+RF	1	0.976	0.981	0.558	0.943	0.926
Bagging+RT	1	0.967	0.974	0.524	0.962	0.902

Table 5.13. Detection rates when using Bagging in Lymphography data set

Method	Class 1	Class 2	Class 3	Class 4
Bagging+NB	0	0.877	0.77	0.25
Bagging+SVM	0	0.926	0.672	0
Bagging+BP	0	0.864	0.459	0
Bagging+RBF	0	0.901	0.787	0.25
Bagging+C4.5	0.5	0.84	0.803	0.75
Bagging+RF	0	0.938	0.77	0.5
Bagging+RT	0	0.938	0.721	0.5

Table 5.14. Detection rates when using Bagging in Glass data set

Method	Class 1	Class 2	Class 3	Class 5	Class 6	Class 7
Bagging+NB	0.8	0.237	0.059	0.308	0.778	0.897
Bagging+SVM	0.814	0.763	0	0.615	0.222	0.759
Bagging+BP	0	0.184	0	0.154	0	0.862
Bagging+RBF	0.671	0.763	0.176	0.462	0.667	0.966
Bagging+C4.5	0.8	0.737	0.235	0.769	1	0.897
Bagging+RF	0.857	0.842	0.235	0.769	0.889	0.828
Bagging+RT	0.814	0.737	0.353	0.692	1	0.724

Table 5.15. Detection rates when using AdaBoost in Intrusion detection data set

Method	Class 1	Class 2	Class 3	Class 4	Class 5
NB	0.99	0.577	0.899	0.44	0.304
SVM	0.68	0.999	0.99	0.4	0.98
BP	0.997	0.994	0.997	0.56	0.975
RBF	0.99	0.99	0.98	0.44	0.92
C4.5	0.94	0.999	1	0.6	0.989
RF	0.997	0.99	0.99	0.24	0.98
RT	0.92	0.84	0.97	0.52	0.4

Table 5.16. Detection rates when using AdaBoost in Thyroid data set

Method	class 1	class 2	class 3
NB	0.808	0.328	0.987
SVM	0	0	1
BP	0.795	0.497	0.992
RBF	0.658	0.011	0.993
C4.5	0.959	0.994	0.995
RF	0.986	1	0.991
RT	0.918	0.927	0.992

Table 5.17. Detection rates when using AdaBoost in Lymphography data set

Method	class 1	class 2	class 3	class 4
NB	0	0.84	0.787	0.75
SVM	0.5	0.852	0.803	0.5
BP	0	0.852	0.754	0.5
RBF	0	0.827	0.787	0.5
C4.5	1	0.864	0.82	0.75
RF	0.5	0.951	0.82	0.75
RT	1	0.753	0.705	0.5

Table 5.18. Detection rates when using AdaBoost in Landsat data set

Method	class 1	class 2	class 3	class 4	class 5	class 7
NB	0.968	0.885	0.933	0.612	0.838	0.81
SVM	1	0	0.026	0	0	0.214
BP	0.973	0.958	0.911	0.586	0.874	0.883
RBF	0.935	0.975	0.948	0.364	0.757	0.842
C4.5	0.968	0.971	0.967	0.544	0.895	0.914
RF	1	0.971	0.981	0.571	0.933	0.908
RT	1	0.947	0.874	0.503	0.8	0.842

Table 5.19. Detection rates when using AdaBoost in Glass data set

Method	Class 1	Class 2	Class 3	Class 5	Class 6	Class 7
NB	0.714	0.197	0.353	0.231	0.889	0.828
SVM	0.729	0.763	0	0.615	0	0.793
BP	0.786	0.724	0.412	0.615	0.667	0.897
RBF	0.786	0.789	0.294	0.615	0.667	0.897
C4.5	0.857	0.803	0.353	0.769	1	0.828
RF	0.871	0.829	0.294	0.769	0.889	0.828
RT	0.757	0.711	0.412	0.846	0.556	0.759

As evident in Tables 5.10- 5.14, we note that using Bagging does not increase the detection rates for minority classes but in reverse using bagging in most data sets decreases the detection rates for other classes which are noticeable when using bagging with SVM and BP classifiers wherever SVM and BP totally biased to the most majority class and became blind from other classes as depicted in figures 5.2-5.6.

About using AdaBoost in Intrusion Detection data set as depicted in Table 5.15 AdaBoost has made insignificant improvement on the performance of all classifiers however; it clearly improves the performance of SVM in the lowest minority class (class 4).Table 5.16 explains the performance of AdaBoost in Thyroid data set, we notice that using AdaBoost does not increase the performance of all classifiers. In Lymphography data set as shown in Table5.17 using AdaBoost increase the detection rates but still NB and SVM are failed to detect class1. In Glass data set as depicted in Table 5.19, using AdaBoost has made a minor improvement especially in the detection rate of class 3 when using with BP but SVM is still blind from class 3 and class 6. Figures 5.2-5.6 explain the effects of using the bases classifiers alone or with Bagging and AdaBoost ensemble in all datasets per class.

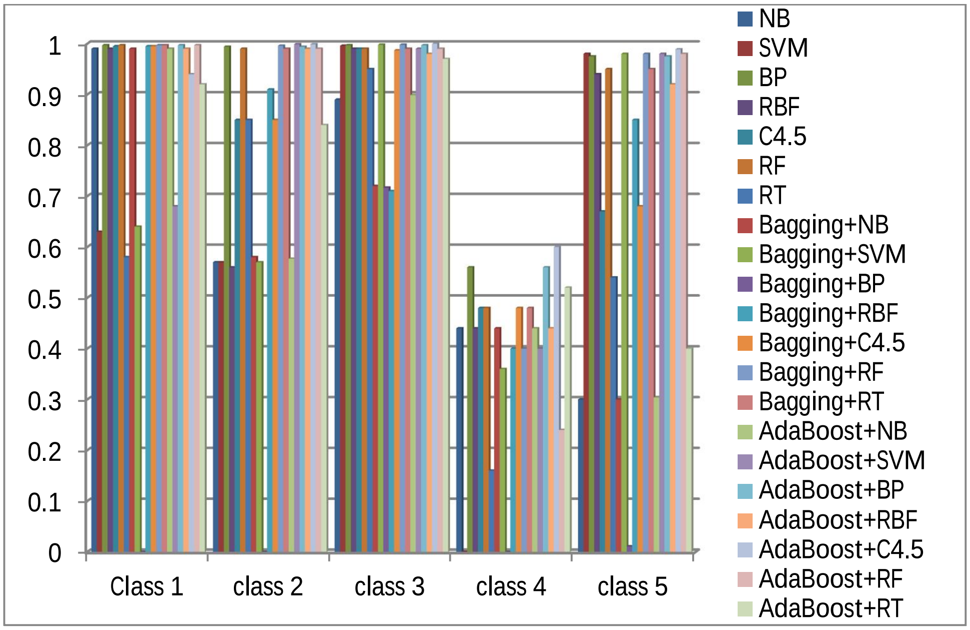

Figure 5.2 Detection rates per class in Intrusion Detection dataset

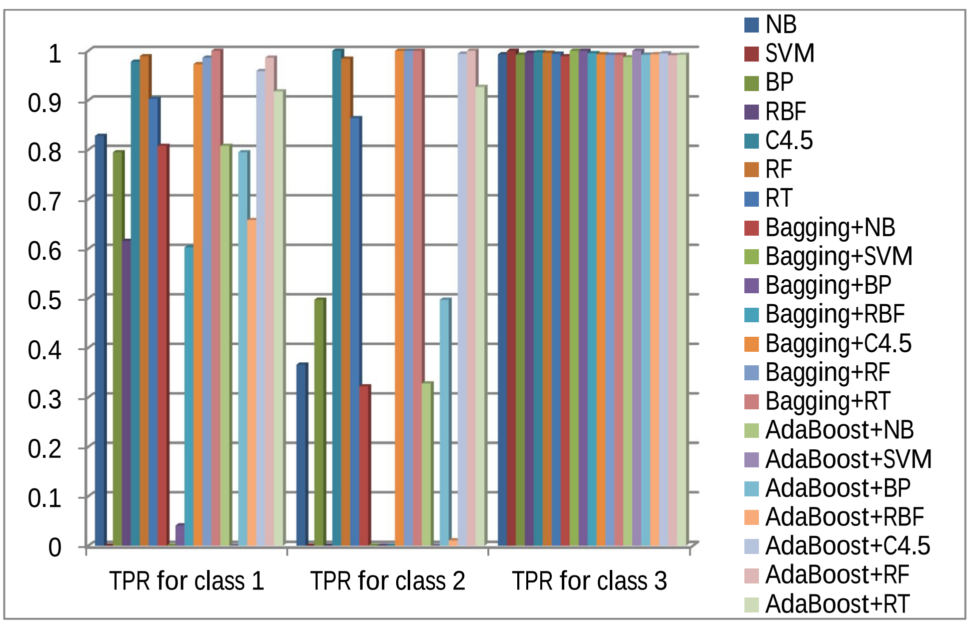

Figure 5.3 Detection rates per class in Thyroid dataset

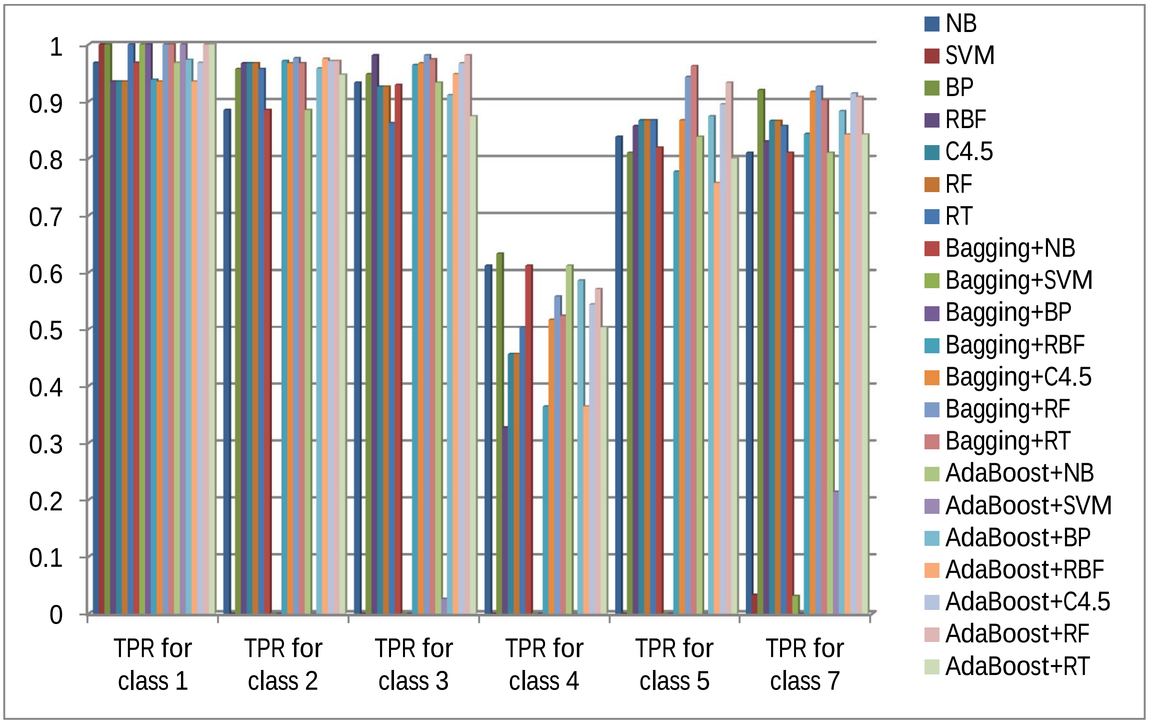

Figure 5.4 Detection rates per class in Land sat dataset

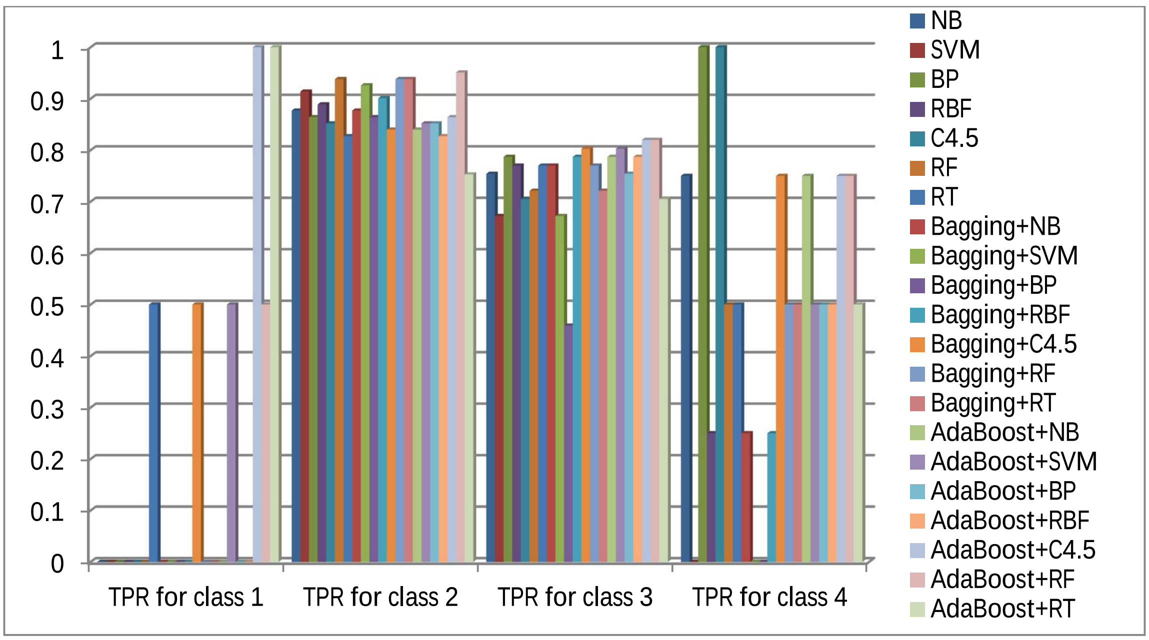

Figure 5.5 Detection rates per class in Lymphography dataset

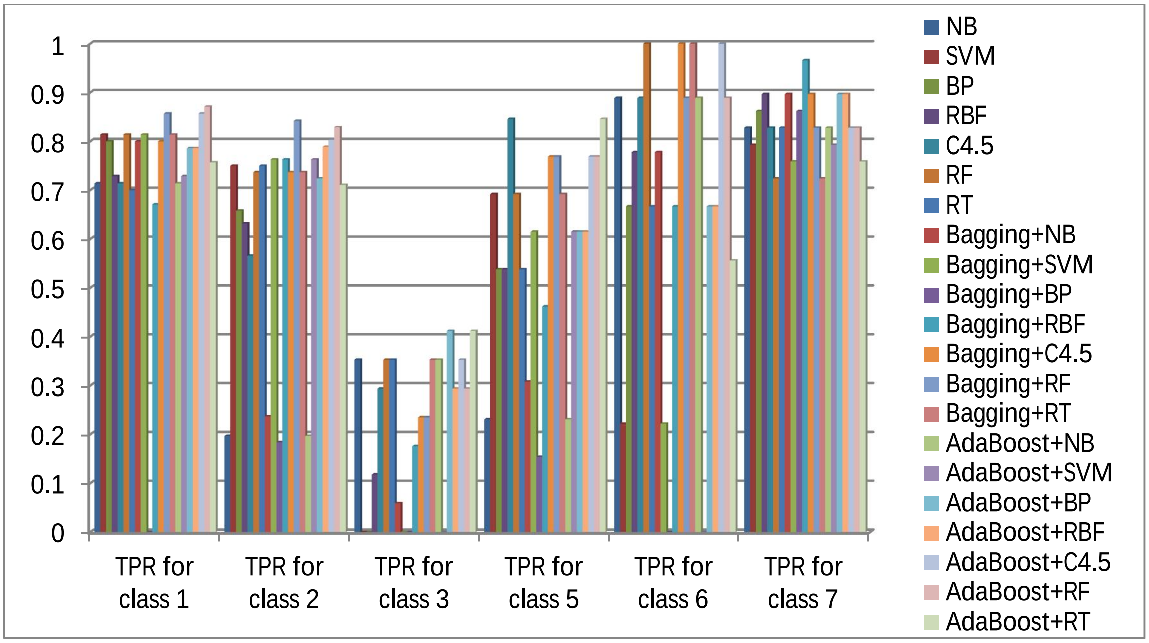

Figure 5.6 Detection rates per class in Glass dataset

5.2.3 Results Analysis and Discussion for Phase Three

To tackle the class's imbalance problem, increase detection rates for each class and minimize false alarms, we propose an ensemble model based on Error-Correct Output Codes (ECOC). In which, the multiclass problem decomposes into several binary sub-problems,and trains a standard classifier for each class. The constructed model must distinguish the samples of a single class (positive class) from all samples in remaining classes (negative class) (See Section 5.1.3). Also here, we tested the performance of seven selected classifiers NB, SVM, BP, RBF, C4.5, RF and RT and compare their result when applied homogenous ensemble methods such as Bagging and AdaBoost for the five classes. For each data set, the m class problem transformed into binary meta sub problems. Then each base classifier is trained to learn specific class. Appendices A-E explains the detailed performance evaluation measures (Recall, Precision and f-measure) for all classifiers for each class meta problem per each data set.

We noticed that using binary classifiers for each Meta binary sub problem perform significant improvements in the detection rates for each class in the binary meta problem. The constructed models for each class are combined together to construct the ECOC ensemble. Then, the predicted class for each testing instance is determined by combined the output of those models to form the codeword. However, there are still misclassifications for those minority classes. So, we noted that in most data sets the performance of using AdaBoost is better than using basic classifiers alone or with Bagging. Hence, to increase the performance of our ECOC ensemble and boost the detection rates of those minority classes, we make a hybrid ensemble that using AdaBoost to learn each classifier for each class. The predicted class is the class with nearest codeword, which is measured by weighted Hamming distance (see Section 5.1.1). Tables 5.20-5.24 explain the performance of the suggested ECOC ensemble per class for each class.

Table5.20. Detection rates per class in Intrusion Detection data set using the proposed hybrid ECOC ensemble

Method	Class 1	Class 2	Class 3	Class 4	Class 5
NB	.995	.994	.993	.92	.992
SVM	.991	1.00	.984	.92	.943
BP	1.00	.996	.998	.96	1.00
RBF	.996	.993	.993	.88	.94
C4.5	1.00	1.00	.936	1.00	1.00
RF	1.00	1.00	.987	1.00	1.00
RT	1.00	1.00	.985	.96	1.00

Table5.21. Detection rates per class in Thyroid data set using the proposed hybrid ECOC ensemble

Method	Class 1	Class 2	Class 3
NB	.968	.995	.995
SVM	.992	.993	.999
BP	.992	.994	.99
RBF	.99	.992	.989
C4.5	1.00	1.00	.998
RF	1.00	1.00	.995
RT	1.00	1.00	.993

Table 5.22. Detection rates per class in Landsat data set using the proposed hybrid ECOC ensemble

Method	Class 1	Class 2	Class 3	Class 4	Class 5	Class 7
NB	.924	.967	.976	.928	.933	.958
SVM	.938	.923	.935	.940	.937	.943
BP	.971	.979	.976	.937	.957	.962
RBF	.943	.981	.922	.904	.911	.921
C4.5	.973	1.00	.978	.976	.979	.993
RF	.96	1.00	.973	.964	.983	.992
RT	.945	1.00	.969	.964	.981	.987

Table5.23. Detection rates per class in Lymphography data set using the proposed ECOC ensemble

Method	Class 1	Class 2	Class 3	Class 4
NB	1.00	.914	.918	1.00
SVM	1.00	.889	.951	1.00
BP	1.00	.852	.951	1.00
RBF	1.00	.852	.918	.75
C4.5	1.00	.889	.97	1.00
RF	1.00	.901	1.00	1.00
RT	1.00	.84	.97	1.00

Table 5.24. Detection rates per class in Glass data set using the proposed hybrid ECOC ensemble

Method	Class 1	Class 2	Class 3	Class 5	Class 6	Class 7
NB	1.00	.914	.941	.942	.992	.974
SVM	1.00	.889	.824	1.00	1.00	.966
BP	1.00	.901	1.00	1.00	1.00	1.00
RBF	1.00	.852	.882	1.00	1.00	1.00
C4.5	1.00	.889	1.00	.981	1.00	1.00
RF	1.00	.901	1.00	1.00	1.00	1.00
RT	1.00	.901	1.00	.981	1.00	1.00

Clearly the results shown in Tables 5.20-5.24 depicted the significant improvements of the proposed ECOC ensemble in the performance. Most of these noticeable improvements are produced by the adaptation of ECOC for class imbalance problem by adding a weight to the Hamming distance function that is equal to 1/ (number of instances in the class). This weight is effective and important for decision when there is an instance has same distance between more than one classes, it gives the higher priority to the minority classes. So, by using this weight we increase the power of our ECOC ensemble by having large codeword distance between any pair of classes and independent bit errors. As a result, we increase the detection rates for each class even those minority classes and minimize false alarms.

5.3. Summary

In this chapter, we investigated the multi imbalanced class problem and compared the performance of three multiclass approaches: the direct multiclass, Bagging and AdaBoost as homogenous ensembles seven of data mining and machine learning algorithms. Also, we proposed a novel hybrid error-correcting code (ECOC) ensemble approach that significantly improves the detection rates for all classes even those minority classes.

CHAPTER SIX

CONCLUSIONS

6.1 Conclusion

The class imbalance is a common problem due to the unequal distribution of data between classes. It is considered as a challenge and critical problem for machine learning and data mining algorithms because the performance would be biased to the majority classes by discarding the minority classes.

In this dissertation, we study the problem of imbalanced class data in both two class data and multi class data using seven of the top data mining and machine learning algorithms and proposed two hybrid ensembles to solve them.

Firstly, we investigated the problem of building models for two class imbalanced data and compared their results when using resampling methods (undersampling and oversampling) and homogenous meta learning methods such as Bagging and AdaBoost. The results revealed that all the tested classifiers are very sensitive for imbalanced classes but the most influenced one is Support Vector Machine, which was biased totally to the negative class and produced True Positive rates equal to zero. Furthermore, employing resampling methods without/with homogenous learning methods improve the performance on the positive class and inversely degrade the performance on the negative class in most cases. This is due to loss of significant and important samples excluded by undersampling and overfitting caused by oversampling. To improve thus problems, we developed a hybrid ensemble approach to improve the performance with an objective to maximize TPRs, precision, f-measure. Our new approach gains the merits for resampling methods and overcome their drawbacks by using both of them at various rates to construct several models for different balanced data sets.

Experimental results on multiple two imbalanced class real data sets with different imbalance ratios, confirms that our approach effectively improve the performance of classifiers in two class imbalance problem especially when the data is highly imbalanced.

Secondly, we investigated the problem of multi class imbalanced data sets and compared their results when using direct multi class classification and homogenous meta learning methods such as Bagging and AdaBoost. Empirical results revealed that Bagging does not increase the detection rates for minority classes but decreased the detection rates for other classes in most data sets mainly when using Support Vector Machine and Back Propagation Neural Network.

Next, we developed a novel Error Correcting Output Code ensemble approach that utilized weighed hamming distance and AdaBoost to train each model in the ensemble. Experimental results on multiple multi imbalanced class real data sets with different imbalance ratios, assures that our novel hybrid approach effectively improve the performance of classifiers in the multi class imbalanced when the data sets are low, moderate or highly imbalanced data. Moreover, the proposed approach significantly improved the classifiers performance even those very sensitive ones by improving the detection rates and decreasing the false alarms.

The proposed hybrid ensembles can be applied to all classifiers since the work is mainly in the preprocessing stage of the data, and this property makes them effective and scalable.

6.2 Future Works

Imbalance class problem is a common problem associated with many real world applications. Hence more studies are definitely needed to continue with many complex problems such as class imbalance with class noise, missing attribute values (attribute noise) in both two and multi class classification problems. Other issues must be taken into account such as small disjuncts and class overlap. Overcoming these problems can be the key for developing new approaches for solving and improving the correct identification of both the minority and majority classes. Also, tackling other data mining and machine learning tasks in class imbalance could be a part of future research.

APPENDICES

APPENDIX A

Table A.1 Performance evaluation measures for class 1 meta problem of Intrusion detection data set

Method	TPR	Precision	f-measure
NB	0.991	0.671	0.8
SVM	0.634	0.824	0.717
BP	0.997	0.979	0.988
RBF	0.972	0.995	0.983
C4.5	0.996	0.978	0.987
RF	0.997	0.996	0.997
RT	0.539	0.943	0.686
Bagging+NB	0.994	0.676	0.805
Bagging+SVM	0.634	0.829	0.716
Bagging+BP	0	0	0
Bagging+RBF	0.995	0.985	0.99
Bagging+C4.5	0.996	0.983	0.989
Bagging+RF	0.997	0.999	0.998
Bagging+RT	0.997	0.996	0.997
AdaBoost+NB	0.991	0.671	0.8
AdaBoost+SVM	0.701	0.835	0.762
AdaBoost+BP	0.997	0.979	0.988
AdaBoost+RBF	0.995	0.937	0.965
AdaBoost+C4.5	0.996	0.998	0.997
AdaBoost+RF	0.996	0.997	0.997
AdaBoost+RT	0.996	0.898	0.944

TableA.2. Performance evaluation measures for class 2 Meta learning problem of Intrusion detection data set

Method	TPR	Precision	f-measure
NB	0.571	0.35	0.434
SVM	0.566	0.995	0.721
BP	0.994	0.993	0.994
RBF	1	0.926	0.962
C4.5	0.993	0.85	0.916
RF	0.853	0.99	0.916
RT	0.987	0.96	0.973
Bagging+NB	0.574	0.423	0.487
Bagging+SVM	0.566	0.995	0.721
Bagging+BP	1	0.123	0.219
Bagging+RBF	0.566	0.983	0.718
Bagging+C4.5	0.994	0.848	0.915
Bagging+RF	0.923	0.998	0.959
Bagging+RT	0.853	0.99	0.916
AdaBoost+NB	0.956	0.868	0.91
AdaBoost+SVM	0.996	0.99	0.993
AdaBoost+BP	0.994	0.993	0.994
AdaBoost+RBF	0.993	0.99	0.998
AdaBoost+C4.5	0.993	0.991	0.992
AdaBoost+RF	0.936	0.989	0.962
AdaBoost+RT	0.986	0.99	0.988

Table A.3. Performance evaluation measures for class 3 meta problem of Intrusion detection data set

Method	TPR	Precision	f-measure
NB	0.93	0.694	0.795
SVM	0.991	0.884	0.9334
BP	0.998	1	0.999
RBF	0.783	0.62	0.692
C4.5	0.996	0.997	0.996
RF	0.98	0.999	0.98
RT	0.936	0.999	0.966
Bagging+NB	0.92	0.695	0.792
Bagging+SVM	0.985	0.884	0.932
Bagging+BP	1	0.995	0.997
Bagging+RBF	0.715	0.714	0.714
Bagging+C4.5	0.996	0.997	0.996
Bagging+RF	0.996	1	0.998
Bagging+RT	0.998	0.999	0.998
AdaBoost+NB	0.993	0.941	0.966
AdaBoost+SVM	0.984	0.892	0.936
AdaBoost+BP	0.998	1	0.999
AdaBoost+RBF	0.716	0.99	0.831
AdaBoost+C4.5	1	0.993	0.996
AdaBoost+RF	0.987	1	0.994
AdaBoost+RT	0.955	0.755	0.843

Table A.4. Performance evaluation measures for class 4 Meta problem of Intrusion detection data set

Method	TPR	Precision	f-measure
NB	.40	.049	.088
SVM	0	0	0
BP	.667	.783	.72
RBF	.333	.643	.439
C4.5	.88	.88	.88
RF	.52	.867	.65
RT	.64	.400	.492
Bagging+NB	.40	.051	.091
Bagging+SVM	.12	.6	.12
Bagging+BP	.333	.643	.439
Bagging+RBF	.04	1.00	.077
Bagging+C4.5	.84	.875	.857
Bagging+RF	.667	1.00	.8
Bagging+RT	.52	.867	.65
AdaBoost+NB	.4	.049	.088
AdaBoost+SVM	.704	.826	.76
AdaBoost+BP	.667	.783	.72
AdaBoost+RBF	.28	.778	.412
AdaBoost+C4.5	.76	.905	.826
AdaBoost+RF	.28	.778	.412
AdaBoost+RT	.6	.75	.667

Table A.5. Performance evaluation measures for class 5 Meta problem of Intrusion detection data set

Method	TPR	Precision	f-measure
NB	0.927	0.737	0.821
SVM	0.922	0.944	0.933
BP	0.679	0.691	0.685
RBF	0.943	0.953	0.948
C4.5	0.92	0.998	0.957
RF	0.655	0.997	0.791
RT	0.992	0.974	0.947
Bagging+NB	0.92	0.698	0.794
Bagging+SVM	0.92	0.942	0.931
Bagging+BP	0.094	1.00	0.172
Bagging+RBF	0.948	0.308	0.465
Bagging+C4.5	0.92	0.998	0.957
Bagging+RF	0.666	0.997	0.799
Bagging+RT	0.655	0.997	0.791
AdaBoost+NB	0.941	0.90	0.92
AdaBoost+SVM	0.94	0.981	0.96
AdaBoost+BP	0.995	0.956	0.975
AdaBoost+RBF	0.911	0.966	0.938
AdaBoost+C4.5	0.943	1.00	0.971
AdaBoost+RF	0.964	0.998	0.981
AdaBoost+RT	0.613	0.969	0.751

APPENDIX B

Table B.1. Performance evaluation measures for class 1 Meta problem of Thyroid data set

Method	TPR	Precision	F-Measure
NB	.882	.739	.804
SVM	0	0	0
BP	.839	.788	.813
RBF	.387	.766	.514
C4.5	.957	.967	.962
RF	.903	.944	.923
RT	.892	.902	.897
Bagging+NB	.828	.819	.824
Bagging+SVM	.71	.846	.772
Bagging+BP	.839	.788	.813
Bagging+RBF	.731	.773	.751
Bagging+C4.5	.452	.875	.596
Bagging+RF	.978	.958	.968
Bagging+RT	.903	.955	.928
AdaBoost+NB	.903	.944	.923
AdaBoost+SVM	.828	.819	.824
AdaBoost+BP	.71	.846	.772
AdaBoost+RBF	.839	.788	.839
AdaBoost+C4.5	.978	.773	.751
AdaBoost+RF	.925	.945	.935
AdaBoost+RT	.903	.913	.908

Table B.2. Performance evaluation measures for class 2 Meta problem of Thyroid data set

Method	TPR	Precision	F-Measure
NB	.012	.209	.153
SVM	0	0	0
BP	.366	.729	.488
RBF	0	0	0
C4.5	1.00	.96	.979
RF	.969	.959	.964
RT	.874	.861	.868
Bagging+NB	.115	.253	.158
Bagging+SVM	0	0	0
Bagging+BP	0	0	0
Bagging+RBF	0	0	0
Bagging+C4.5	1.00	.965	.982
Bagging+RF	.979	.979	.977
Bagging+RT	.969	.959	.964
AdaBoost+NB	.44	.683	.535
AdaBoost+SVM	0	0	0
AdaBoost+BP	.366	.729	.488
AdaBoost+RBF	.016	1.00	.031
AdaBoost+C4.5	.995	.974	.984
AdaBoost+RF	.969	.959	.964
AdaBoost+RT	.843	.861	.852

Table B.3. Performance evaluation measures for class 3 Meta learning problem of Thyroid data set

Method	TPR	Precision	F-Measure
NB	.993	.991	.992
SVM	1.00	.925	.961
BP	.99	.979	.984
RBF	.998	.929	.963
C4.5	.997	.999	.998
RF	.995	.999	.997
RT	.991	.993	.992
Bagging+NB	.993	.959	.976
Bagging+SVM	1.00	.925	.961
Bagging+BP	.995	.933	.963
Bagging+RBF	1.00	.925	.961
Bagging+C4.5	.996	.999	.998
Bagging+RF	.997	1.00	.998
Bagging+RT	.995	.999	.997
AdaBoost+NB	.995	.957	.975
AdaBoost+SVM	.999	.932	.964
AdaBoost+BP	.99	.979	.984
AdaBoost+RBF	.984	.955	.969
AdaBoost+C4.5	.998	1.00	.997
AdaBoost+RF	.995	1.00	.997
AdaBoost+RT	.993	.991	.992

APPENDIX C

Table C.1. Performance evaluation measures for class 1 Meta problem of Lymphography data set

Method	TPR	Precision	F-Measure
NB	1.00	.333	.50
SVM	0	0	0
BP	1.00	.667	.8
RBF	.50	.50	.50
C4.5	0	0	0
RF	0	0	0
RT	.5	1.00	.667
Bagging+NB	0	0	0
Bagging+SVM	0	0	0
Bagging+BP	0	0	0
Bagging+RBF	0	0	0
Bagging+C4.5	0	0	0
Bagging+RF	0	0	0
Bagging+RT	.50	1.00	.667
AdaBoost+NB	0	0	0
AdaBoost+SVM	0	0	0
AdaBoost+BP	1.00	.667	.8
AdaBoost+RBF	0	0	0
AdaBoost+C4.5	0	0	0
AdaBoost+RF	1.00	1.00	1.00
AdaBoost+RT	.50	.50	.50

Table C.2. Performance evaluation measures for class 2 Meta problem of Lymphography data set

Method	TPR	Precision	F-Measure
NB	.889	.837	.862
SVM	.914	.771	.836
BP	.852	.841	.847
RBF	.852	.841	.847
C4.5	.864	.843	.854
RF	.84	.872	.855
RT	.778	.768	.773
Bagging+NB	.914	.841	.876
Bagging+SVM	.914	.771	.836
Bagging+BP	.889	.837	.862
Bagging+RBF	.901	.849	.837
Bagging+C4.5	.852	.863	.857
Bagging+RF	.926	.824	.872
Bagging+RT	.84	.872	.855
AdaBoost+NB	.914	.841	.876
AdaBoost+SVM	.889	.828	.857
AdaBoost+BP	.852	.841	.847
AdaBoost+RBF	.852	.852	.852
AdaBoost+C4.5	.889	.847	.867
AdaBoost+RF	.901	.88	.89
AdaBoost+RT	.753	.753	.753

Table C.3. Performance evaluation measures for class 3 Meta problem of Lymphography data set

Method	TPR	Precision	F-Measure
NB	.852	.852	.852
SVM	.672	.872	.759
BP	.77	.825	.797
RBF	.787	.842	.814
C4.5	.738	.763	.75
RF	.721	.846	.779
RT	.686	.724	.706
Bagging+NB	.82	.877	.847
Bagging+SVM	.705	.843	.768
Bagging+BP	.77	.87	0.817
Bagging+RBF	.82	.847	.833
Bagging+C4.5	.803	.754	.778
Bagging+RF	.738	.9	.811
Bagging+RT	.721	.846	.779
AdaBoost+NB	.803	.817	.81
AdaBoost+SVM	.721	.772	.746
AdaBoost+BP	.77	.825	.797
AdaBoost+RBF	.803	.817	.81
AdaBoost+C4.5	.738	.763	.75
AdaBoost+RF	.738	.818	.776
AdaBoost+RT	.738	.703	.72

Table C.4. Performance evaluation measures for class 4 Meta problem of Lymphography data set

Method	TPR	Precision	F-Measure
NB	1.00	.571	.727
SVM	0	0	0
BP	.75	1	.857
RBF	1.00	.667	.8
C4.5	.75	1.00	.857
RF	.50	1.00	.50
RT	.75	0.60	.667
Bagging+NB	1.00	.571	.727
Bagging+SVM	0	0	0
Bagging+BP	0	0	0
Bagging+RBF	0	0	0
Bagging+C4.5	.50	1.00	.667
Bagging+RF	.75	1.00	.857
Bagging+RT	.50	1.00	.667
AdaBoost+NB	.75	1.00	.857
AdaBoost+SVM	.50	.667	.571
AdaBoost+BP	1.00	1.00	1.00
AdaBoost+RBF	0	0	0
AdaBoost+C4.5	.75	1.00	.857
AdaBoost+RF	.75	1.00	.857
AdaBoost+RT	.75	.60	.667

Table D.1. Performance evaluation measures for class 1 Meta learning problem of Landsat data set

Method	TPR	Precision	F-Measure
NB	.833	.892	.862
SVM	0	0	0
BP	.971	.982	.977
RBF	.94	.936	.938
C4.5	.938	.955	.947
RF	.952	.98	.966
RT	.945	.938	.941
Bagging+NB	.831	.891	.86
Bagging+SVM	0	0	0
Bagging+BP	0	0	0
Bagging+RBF	.942	.94	.941
Bagging+C4.5	.951	.972	.961
Bagging+RF	.958	.975	.967
Bagging+RT	.952	.98	.966
AdaBoost+NB	.924	.95	.924
AdaBoost+SVM	.938	.957	.947
AdaBoost+BP	.971	.982	.977
AdaBoost+RBF	.943	.947	.945
AdaBoost+C4.5	.973	.983	.978
AdaBoost+RF	.96	.983	.971
AdaBoost+RT	.945	.949	.947

Table D.2. Performance evaluation measures for class 2 Meta learning problem of Landsat data set

Method	TPR	Precision	F-Measure
NB	.90	.966	.90
SVM	0	0	0
BP	.965	.973	.969
RBF	.981	.85	.911
C4.5	.944	.966	.955
RF	.947	.995	.971
RT	.939	.949	.944
Bagging+NB	.90	.966	.932
Bagging+SVM	0	0	0
Bagging+BP	0	0	0
Bagging+RBF	.979	.856	.913
Bagging+C4.5	.965	.967	.966
Bagging+RF	.962	.971	.962
Bagging+RT	.954	.974	.964
AdaBoost+NB	.923	.957	.939
AdaBoost+SVM	0	0	0
AdaBoost+BP	.965	.973	.969
AdaBoost+RBF	.96	.966	.963
AdaBoost+C4.5	.967	.977	.972
AdaBoost+RF	.967	.979	.973
AdaBoost+RT	.946	.946	.946

Table D.3. Performance evaluation measures for class 3 Meta learning problem of Landsat data set

Method	TPR	Precision	F-Measure
NB	.956	.882	.917
SVM	0.024	1.00	0.047
BP	.964	.938	.951
RBF	.96	.898	.928
C4.5	.953	..933	.943
RF	.969	.948	.959
RT	.94	.931	.936
Bagging+NB	.956	.882	.918
Bagging+SVM	0.014	0	0.027
Bagging+BP	0	0	.0
Bagging+RBF	.957	.896	.925
Bagging+C4.5	.976	.94	.958
Bagging+RF	.983	.944	.963
Bagging+RT	.969	.948	.959
AdaBoost+NB	.968	.897	.931
AdaBoost+SVM	.17	1.00	.291
AdaBoost+BP	.964	.938	.951
AdaBoost+RBF	.922	.918	.92
AdaBoost+C4.5	.978	.978	.691
AdaBoost+RF	.973	.95	.961
AdaBoost+RT	.932	.938	.935

TableD.4. Performance evaluation measures for class 4 Meta learning problem of Landsat data set

Method	TPR	Precision	F-Measure
NB	.876	.492	.63
SVM	.024	1.00	0.047
BP	.807	.798	.802
RBF	.601	.535	.566
C4.5	.718	.727	.722
RF	.729	.879	.797
RT	.737	.725	.731
Bagging+NB	.872	.492	.629
Bagging+SVM	0	0	0
Bagging+BP	0	0	0
Bagging+RBF	.577	.562	.569
Bagging+C4.5	.746	.844	.792
Bagging+RF	.748	.875	.806
Bagging+RT	.729	.879	.797
AdaBoost+NB	.876	.492	.63
AdaBoost+SVM	.057	1.00	.107
AdaBoost+BP	.81	.801	.805
AdaBoost+RBF	.792	.855	.822
AdaBoost+C4.5	.802	.912	.854
AdaBoost+RF	.746	.724	.735
AdaBoost+RT	.81	.801	.805

Table D.5. Performance evaluation measures for class 5 Meta learning problem of Landsatdata set

Method	TPR	Precision	F-Measure
NB	.829	.434	.569
SVM	.002	1.00	0.004
BP	.937	.922	.929
RBF	.534	.746	.622
C4.5	.914	.916	.915
RF	.911	.975	.942
RT	.891	.886	.889
Bagging+NB	.827	.433	.568
Bagging+SVM	.001	1.00	.002
Bagging+BP	0	0	0
Bagging+RBF	.534	.746	.622
Bagging+C4.5	.914	.954	.934
Bagging+RF	.933	.973	.953
Bagging+RT	.911	.975	.942
AdaBoost+NB	.829	.434	.569
AdaBoost+SVM	0.015	1.00	0.029
AdaBoost+BP	.937	.922	.929
AdaBoost+RBF	.812	.853	.832
AdaBoost+C4.5	.941	.959	.95
AdaBoost+RF	.957	.975	.966
AdaBoost+RT	.886	.876	.881

Table D.6. Performance evaluation measures for class 7 Meta learning problem of Landsat data set

Method	TPR	Precision	F-Measure
NB	.891	.813	.85
SVM	.108	1.00	.195
BP	.946	.924	.935
RBF	.963	.833	.893
C4.5	.921	.912	.917
RF	0.943	.941	.942
RT	.921	.91	.916
Bagging+NB	.891	.813	.85
Bagging+SVM	.072	1.00	.134
Bagging+BP	0	0	0
Bagging+RBF	.962	.836	.894
Bagging+C4.5	.953	.934	.943
Bagging+RF	.962	.942	.952
Bagging+RT	.943	.941	.942
AdaBoost+NB	.898	.824	.86
AdaBoost+SVM	.146	1.00	.255
AdaBoost+BP	.946	.924	.935
AdaBoost+RBF	.897	.882	.889
AdaBoost+C4.5	.958	.944	.951
AdaBoost+RF	.963	.956	.959
AdaBoost+RT	.921	.91	.916

Table E.1. Performance evaluation measures for class 1 Meta learning problem of Glass data set

Method	TPR	Precision	F-Measure
NB	.975	.599	.742
SVM	.916	.746	.822
BP	.911	.844	.876
RBF	.871	.669	.757
C4.5	.911	.818	.862
RF	.911	.915	.913
RT	.871	.876	.873
Bagging+NB	.975	.602	.745
Bagging+SVM	.916	.749	.824
Bagging+BP	0	0	0
Bagging+RBF	.851	.696	.766
Bagging+C4.5	.916	.889	.902
Bagging+RF	.931	.913	.922
Bagging+RT	.911	.915	.913
AdaBoost+NB	.975	.599	.742
AdaBoost+SVM	.95	.715	.816
AdaBoost+BP	.916	.853	.883
AdaBoost+RBF	.856	.816	.836
AdaBoost+C4.5	.936	.9	.917
AdaBoost+RF	.941	.936	.938
AdaBoost+RT	.896	.887	.892

Table E.2. Performance evaluation measures for class 2 Meta learning problem of Glass data set

Method	TPR	Precision	F-Measure
NB	.941	.636	.759
SVM	.862	.708	.777
BP	0.809	.778	.794
RBF	.862	.686	.764
C4.5	.776	.825	.80
RF	.862	.832	.856
RT	.882	.832	.856
Bagging+NB	.941	.636	.759
Bagging+SVM	.849	.713	.775
Bagging+BP	0	0	0
Bagging+RBF	.875	.689	.771
Bagging+C4.5	.855	.844	.855
Bagging+RF	.868	.857	.863
Bagging+RT	.862	.885	.873
AdaBoost+NB	.868	.663	.752
AdaBoost+SVM	.809	.715	.759
AdaBoost+BP	.849	.849	.827
AdaBoost+RBF	.809	.715	.759
AdaBoost+C4.5	.855	.878	.867
AdaBoost+RF	.901	.938	.919
AdaBoost+RT	.829	.773	.8

Table E.3. Performance evaluation measures for class 3 Meta learning problem of Glass data set

Method	TPR	Precision	F-Measure
NB	.882	.355	.506
SVM	0	0	0
BP	.765	.788	.776
RBF	.735	.588	.654
C4.5	.853	.866	.859
RF	.838	.905	.87
RT	.721	.817	.766
Bagging+NB	.882	.361	.513
Bagging+SVM	0	0	0
Bagging+BP	1.00	.258	.41
Bagging+RBF	.618	.689	.651
Bagging+C4.5	.853	.879	.866
Bagging+RF	.853	.892	.872
Bagging+RT	.838	.905	.87
AdaBoost+NB	.882	.355	.506
AdaBoost+SVM	.618	.525	.568
AdaBoost+BP	.824	.80	.812
AdaBoost+RBF	.838	.781	.809
AdaBoost+C4.5	.838	.838	.838
AdaBoost+RF	.853	.921	.885
AdaBoost+RT	.894	.778	.80

Table E.4. Performance evaluation measures for class 5 Meta learning problem of Glass data set

Method	TPR	Precision	F-Measure
NB	.712	.755	.733
SVM	1	.912	.954
BP	.981	.836	.903
RBF	.904	.87	.887
C4.5	.923	.889	.906
RF	.923	.96	.941
RT	.885	.939	.911
Bagging+NB	.692	.8	.742
Bagging+SVM	.942	.907	.925
Bagging+BP	0	0	0
Bagging+RBF	.885	.92	.902
Bagging+C4.5	.942	.925	.933
Bagging+RF	.981	.962	.971
Bagging+RT	.923	.96	.941
AdaBoost+NB	.923	.873	.897
AdaBoost+SVM	1	.912	.954
AdaBoost+BP	.981	.836	.903
AdaBoost+RBF	.942	.961	.951
AdaBoost+C4.5	.942	.875	.907
AdaBoost+RF	.942	.961	.951
AdaBoost+RT	.904	.904	.904

Table E.5. Performance evaluation measures for class 6 Meta learning problem of Glass data set

Method	TPR	Precision	F-measure
NB	0.972	0.897	0.933
SVM	0.972	0.875	0.921
BP	1	0.9	0.947
RBF	0.944	0.944	0.944
C4.5	1.00	0.973	0.986
RF	0.944	1.00	0.971
RT	1.00	0.973	0.986
Bagging+NB	0.972	0.875	0.921
Bagging+SVM	0.972	0.875	0.921
Bagging+BP	0	0	0
Bagging+RBF	0.944	0.944	0.944
Bagging+C4.5	1.00	0.947	0.973
Bagging+RF	1.00	0.973	0.986
Bagging+RT	0.944	1.00	0.971
AdaBoost+NB	0.972	0.921	0.946
AdaBoost+SVM	0.972	0.921	0.946
AdaBoost+BP	1.00	0.9	0.947
AdaBoost+RBF	0.944	0.971	0.958
AdaBoost+C4.5	0.972	0.946	0.959
AdaBoost+RF	0.944	1.00	0.971
AdaBoost+RT	0.944	0.944	0.944

Table E.6. Performance evaluation measures for class 7 Meta learning problem of Glass data set

Method	TPR	Precision	F-Measure
NB	.922	.939	.93
SVM	.94	.973	.956
BP	.948	.948	.948
RBF	.957	.965	.961
C4.5	.931	.90	.915
RF	.94	.956	.948
RT	.957	.902	.929
Bagging+NB	.922	.947	.934
Bagging+SVM	.94	.982	.96
Bagging+BP	.284	.465	.353
Bagging+RBF	.94	.948	.944
Bagging+C4.5	.948	.965	.957
Bagging+RF	.94	.956	.948
Bagging+RT	.905	.921	.913
AdaBoost+NB	.951	.921	.913
AdaBoost+SVM	.957	.982	.969
AdaBoost+BP	.966	.933	.949
AdaBoost+RBF	.974	.958	.966
AdaBoost+C4.5	.966	.941	.953
AdaBoost+RF	.948	.973	.961
AdaBoost+RT	.948	.94	.944

References

[1] O. Maimon and L. Rokach, *Data Mining and Knowledge Discovery Handbook*, 2nd ed.: Springer, 2010.

[2] N. Satuluri and M. R. Kuppa, "A novel Class Imbalance Learning Using Intelligent Under-Sampling ," *international Journal of Database Theory and Application*, vol. 5, pp. 25-35, 2012.

[3] A. Bouzerdoum and G. Nguyen S. Phung, "Learning Patterns classification tasks with imbalanced data sets," in *Pattern Recognition*. Vukovar, Croatile: In-Tech, 2009, ch. 10, pp. 193-208.

[4] M. S. Kamel, A. K.C.Wong and Y. Wang Y. Sun, "Cost-sensitive Boosting for Classification of Imbalanced Data," *Pattern Recognition*, vol. 40, pp. 3358-3378, 2007.

[5] J. S. Sanchez and R. A. Mollineda V. Garcia, "On the Effectiveness of Preprocessing Methods when Dealing with Different Levels of Class Imbalance," *Knowledge-Based Systems* , vol. 25, pp. 13-21, 2012.

[6] R. Prati and M. Monard G. Batista, "A Study of the Behaviour of Several methods for balancing Machine Learning Training Data," *SIGKDD*, vol. 6, no. 1, pp. 20-29, 2004.

[7] I. Tomek, "Two Modifications of CNN," *IEEE Transaction on Systems and Man and Sybernetics*, vol. 6, pp. 769-772, 1976.

[8] P. Hart, "The Condenced nearest neighbour rule," *IEEE Transactions on Information Theory* , vol. 14, no. 5, pp. 515-516, 1968.

[9] M. Kubat and S. Matwin, "Addressing the Curse of Imbalanced Training Data Sets: One Sided Selection," in *14th International Conference on Machine Learning(ICML97)*, 1997, pp. 179-186.

[10] Y. Wang and S. H. Bryant Q. Li, "A novel Method for Mining Highly Imbalanced high-throughput Screening Data in PubChem," *Bioinformatics*, vol. 25, pp. 3310-3316, 2009.

[11] R. M. Valdovinos, J. S. Sanchez and F. J. Ferri R. Barandela, "The Imbalanced Training Sample Problem: Under or Over Sampling," in *Joint IAPR International Workshops on Structural, Syntactic and Statistical Pattern Recognition (SSPR/SPR04)*, 2004, pp. 806-814.

[12] Yen and Y. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," *Expert Systems with Applications*, vol. 36, pp. 5718-5727, 2009.

[13] M. M. Rahman and D. N. Davis, "Addressing the Class Imbalance Problem in Medical Datasets," *International Journal of Machine Learning and Computing*, vol. 3, no. 2, pp. 224-228, April 2011.

[14] K. W. Bower, L. O. Hall and W. P. Kegelemer N. V. Chawla, "SMOTE: Synthetic Minority Over-Sampling Technique," *Artificial Intelligence Research*, vol. 16, pp. 321-357, 2002.

[15] Z. Dongemi and L. Yang L. Wei, "A novel Over-Sampling Method Based on EDAs for Learning from Imbalanced Data Sets," *Journal of convergence Information Technology*, vol. 6, no. 11, pp. 237-247, 2011.

[16] E. W. Cooper and K. Kamei H. M. Nguyen, "Borderline Over-sampling for Imbalanced Classification," in *Fifth International coference on Computational Intelligence & Applications*, 2009, pp. 24-29.

[17] D. Juan and J. Li-li, "The Research for KNN and SVM Classification Performance on Two Kinds of Unbalanced Data Set," in *Proceedings of the 2nd Conference on Computer Science and Electronic Engineering (ICCSEE)*, 2013.

[18] H. S. Yasdi and R. Menseli A. GhaziKhani, "Class Imbalance Handling Using Wrapper-based Random Oversampling," in *20th Iranian Conference on Electrical Engineering*, 2012, pp. 611-616.

[19] A. Amonden, T. Kakimoto and K. Matsumoto Y. Kamei, "The Effects of Over and Under Sampling on Fault-Prone Moodule Detection," *IEEE*, pp. 196-204, 2007.

[20] On the Generation of Accurate Predictive Model from Highly Imbalanced Data with Heuristics and Replication Technologies, "N. Kerdprasop and K. Kerdprasop ," *International Journal of Bio-Science and Bio-Technology*, vol. 4, pp. 49-64, 2012.

[21] N. Verbiest, Y. Caballero and C. Cornelis E. Ramentol, "SMOTE-FRST: A New Resampling Method Using Fuzzy Rough Set Theory," in *WSPC*, 2012.

[22] A. M. Din and K. R. Ku-Mahamud M. Zorketlee, "Fuzzy and SMOTE Resampling Technique for Imbalanced Data Set," in *Proceedings of the 5th International Conference on Computing and Informatics*, Istanbul, 2015, pp. 638-643.

[23] M. Vasu and V. Ravi, "A Hybrid Under-sampling Approach for Mining Unbalanced Datasets: Applications to Banking and Insurance," *International Journa ofl Data modeling and Management*, vol. 3, pp. 75-105, 2011.

[24] Y. Mi, "Imbalanced Classification Based on Active Learning SMOTE," *Research Journal of Applied Science Engineering and Technology*, vol. 5, pp. 944-949, 2013.

[25] A. Biodgloi and M. N. Parsa, "A Hyprid Feature Selection by Resampling, Chi squared and Consistency Evaluation Techniques," *World Academy of Science, Engineering and Technology*, vol. 68, 2012.

[26] A. O. Puig and E.B. Mansilla, "Evolutionary rule-based systems for imbalanced data sets," *Soft Computing*, vol. 13, pp. 213-225, 2009.

[27] S. Hashemi and A. Hamzeh M. Alibeigi, "Unsupervised Feature Selection Based on the Distribution of Features Attributes to Imbalanced Data Sets," *International Journal of Artificial Intelligence and Expert Systems (IJAE)*, vol. 9, no. 1, pp. 14-22, 2011.

[28] A. Muñoz-Meléndez, E. F. Morales G. Cuaya, "A Minority Class Feature Selection Method," *in Progress in Pattern Recognition, Image Analysis, Computer Vision, and Application*, pp. 417-424.

[29] J. Guo, W. Xu and K. Yao F. Chang, "A Feature Selection Method to Handle Imbalanced Data in Text Classification," *Journal of Digital Information Management*, vol. 13, no. 3, pp. 169-175, june 2015.

[30] T. M. Khoshgoftaar and K. Gap, "Feature Selection with Imbalanced Data for Defect Prediction," in *International Conference on Machine Learning and Applications*, 2009, pp. 235-240.

[31] K. Gao and V. Hulse T. M. Khoshgoftaar, "A Novel Feature Selection Technique for Highly Imbalanced Data," in *International Conference on Information Reuse*, 2010, pp. 80-85.

[32] D. R. Lakshmi and T. V. Rao K. N. Rao, "A Novel Class Imbalance Learning Method using Neural Networks," *International Journal of Applied Information Systems*, vol. 3, no. 7, pp. 31-38, August 2012.

[33] T. V. Rao and D. R. Lakshmi K. N. Rao, "A Novel Class Imbalance Learning Method using Subset Filtering," *International Journal of Scientific & Engineering Research*, vol. 3, no. 9, pp. 4-9, September 2012.

[34] N. SAtuluri and M. R. Kuppa, "A Novel Class Imbalance Learning Using Intelligent Under-Sampling," *International Journal of Database Theory and Application*, vol. 5, no. 3, pp. 25-35, September 2012.

[35] S. Naganjaneyulu and M. R. Kuppa, "A novel framework for class imbalance learning using intelligent Under-sampling," *Prog Artif Intel*, vol. 2, pp. 73-84, 2013.

[36] M. S. Kamel, A. K.C. Wong and Y. Wang Y. Sun, "Cost Sensitive Boosting for Classification of Imbalanced Data," *Pattern Recognition*, vol. 40, pp. 3358-3378, 2007.

[37] A. Bener, H. N. Ciracy and M. Bahceci A. Uyar, "Handling the Imbalance Problem of IVF Implantation Prediction," *IAENG International Journal of Computer Science*, 2006.

[38] P. Rojanavasu and O. Pinngern, " , IEEE, pp. 132-136, 2008 N. H. Thach, "Cost-sensitive XCS Classifier System Addressing Imbalance Problems," in *Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, 2008, pp. 132-136.

[39] R.M. Valdovinos, V. Garcia and J.H. Pacheco-Sanchez R. Alejo, "A hyprid method to face class overlap and class imbalance on neural networks and multi-class scenario," *Pattern Recognition Letters*, vol. 34, pp. 3880-3888, 2013.

[40] X.g Liu and Z. Zhou, "The Influence of Class Imbalance on Cost-Sensitive Learning: An Empirical Study ," in *sixth International Conference on Data Mining*, 2006, pp. 970-974.

[41] E. Lim and Y. Liu A. Sun, "On Strategies for Imbalanced Text Classifcation Using SVM:An Empirical Study," *Decision Support Systems*, 2009.

[42] X. Hong, S. Chen and C. Harris M. Gao, "A combined SMOTE and PSO based RBF Classifier for Two-class Imbalanced Problems," *Neuro Computing*, vol. 74, pp. 3456-3466, 2011.

[43] M. Daneshmandi and M. Ahmadzadeh, "A Hybrid Data Mining Model to Improve Customer Response Modeling in Direct Marketing," *Indian Journal of Computer Science and Engineering (IJCSE)*, vol. 3, no. 6, pp. 844-855, Dec 2013.

[44] Z. Shapiai and M. Khalid A. Adam, "A Modified Artificial Neural Networks Learning Algorithm for Imbalanced Data Set Problem," in *Second Intelligence, Communication Systems and Networks (CICYN)*, 2010, pp. 44-48.

[45] D. Aouada and B. Ottersten A. C. Bahnsen, "A novel Cost-Sensitive Framework for Customer Churn Predictive Modeling," *Decision Analytics*, pp. 1-15, 2015.

[46] R. Kothandan, "Handling class imbalance problem in miRNA dataset associated with cancer," *Bioinformation*, vol. 11, no. 1, pp. 6-10, 2015.

[47] C. X. Ling and V. S. Sheng, "Cost-Sensitive Learning and the Class Imbalance Problem," *Encyclopedia of Machine Learning*, 2008.

[48] A. Bener, H. N. Ciracy and M. Bahcec A. Uyar, "Handling the Imbalance Problem of IVF Implantation Prediction," *IAENG International Journal of Computer Science*, 2006.

[49] D. Xie and Z. Du L. Yan, "A new Method of Support vector Machine for Class Imbalance Problem," in *International Joint Conference on Computational Science and Optimization*, 2009, pp. 904-907.

[50] B. Raskutti and A. Kowalczyk, "Extreme Re-balancing for SVMs: a case study," *SIGKDD Exploration*, vol. 6, pp. 60-69, 2004.

[51] T. Kim and H. Ahn, "A Hybrid Under-sampling Approach for Better Bankruptcy Prediction," *J Intell Syst* , vol. 21, no. 2, pp. 173-190, June 2015.

[52] L. Zhuang and H. Dai, "Parameter Optimization of Kernel-based One-class Classifier on Imbalance Learning," *journal of Computers*, vol. 1, no. 7, October/November 2006.

[53] A. Fernando, E. Barrenechea, H. Business and F. Herrera M. Galar, "A Review on ensembles for the class Imbalance Problem," *IEEE Transactions on Systems, Man and Cybernetics- Part C: Applications and Reviews*, vol. 42, 2012.

[54] P. Kang and S. Cho, "EUS SVMs: Ensemble of Under- Sampled SVMs for Data Imbalance Problems," in *International Conference on Neural Information Processing*, 2006.

[55] Y. Zhang and D. Wang, "A Cost-Sensitive Ensemble Method for Class-Imbalanced Datasets," *Introduction and Applied Analysis*, 2013.

[56] T. M. Khoshgoftaar, J. V. Hulse and A. Napolitano C. Seiffert, "Building Useful Models from Imbalanced with Sampling and Boosting," in *Proceedings of The Twenty-First International FLAIRS Coference*, 2008, pp. 306-311.

[57] K. L. Chan and W Fang P. Li, "Hybrid Kernel Machine Ensemble for Imbalanced Data Sets ," in *18th International Conference on Pattern Recognition*, 2006, pp. 1108-1111.

[58] T. M. Khoshgoftaar, J. V. Hulse and A. Napolitano C. Seiffert, "RUSBoost: A Hybrid Approach to Alleviating Class Imbalance," *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, vol. 40, no. 1, pp. 185-197.

[59] M. Govindaraj and S. Lavanya, "A Combined Boosting And Sampling Approach For Imbalanced Data Classification," *International Journal of Advanced Research in Data Mining and Cloud Computing*, vol. 1, no. 1, pp. 44-50, July 2013.

[60] B. Yuan and X. Ma, "Sampling + Reweighting: Boosting the Performance of AdaBoost on Imbalanced Datasets," *IEEE World Congress on Computational Intelligence*, pp. 2680-2685, June 2012.

[61] Y. Ren and P. Jia H. Xiong, "A Novel Classification Approach for C2C E-Commerce Fraud Detection," *International Journal of Digital Content Technology and its Applications*, vol. 7, no. 1, pp. 504-511, January 2013.

[62] J. Wu and Z. Zhou X. Liu, "Exploratory Undersampling for Class-Imbalance Learning," *IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS – PART B 1*, vol. 39, no. 2, pp. 500-539, 2008.

[63] L. Tianyu, "Imbalance learning for fault diagnosis gearbox in wind turbine," *Journal of Chemical and Pharmaceutical Research*, vol. 7, no. 3, pp. 1287-1292, 2015.

[64] T. M. KHOSHGOFTAAR and R. WALD K. GAO, "T he Use of Under- and Oversampling within Ensemble Features Selection and Classification of Software Quality Prediction," *International Journal of Reliability, Quality and Safety Engineering*, vol. 21, no. 1, 2014.

[65] J. Burez and D. V. Poel, "Handling class imbalance in customer churn prediction," *Experts System with Applications*, vol. 36, pp. 4626-4636, 2009.

[66] H. Gue and H. Viktor, "Learning from Imbalanced Data Sets with Boosting and Generation: The DataBoost-IM Approach," *SIGKDD Explorations*, vol. 6, pp. 30-39.

[67] Y. Zhang and B. Luo, "Parallel Classification Ensemble with Hierarchical Machine Learning for Imbalanced Classes," in *The seventh International conference on Machine Learning and Cybernetics*, Kunming, 2008.

[68] (2011) Medical Informatic & Decision Making. [Online]. http://www.biomedcentral.com/1472-6947/11/51

[69] Q. Xu and L. Zhou H. Wang, "Seminal Quality Prediction Using Clustering-Based Decision Forests," *Algorithms*, vol. 7, pp. 405-417, 2014.

[70] Michal Wo´zniak and G. Schaefer B. Krawczyk, "Cost Sensitive Decision Tree Ensembles for Effective imbalanced Classification," *Applied Soft Computing*, pp. 554-562.

[71] H. Gupta M. Sahare, "A Review of Multi-Class Classification for Imbalanced Data," *International Journal of Advanced Computer Research* , vol. 2, no. 3, pp. 163-168, September 2012.

[72] M. Aly, "Survey on multi-class classification methods," Caltech, USA, Technical Report 2005.

[73] M. F. Al-Roby and A. M. El-Halees, "Classifying Muti-Class Imbalance Data," *Egyptian Computer Science Journal*, vol. 37, no. 5, pp. 74-81, September 2013.

[74] H. Wang, G. Ou and L. A. Feldkamp Y. L. Murphy, "OAHO: An Effective Algorithm for Multi-class Learning from Imbalanced Data," in *International Joint Conference on Neural Networks*, 2007, pp. 406-411.

[75] "Enhansing Classifier Performance of Multiclass Imbalanced Data Using the OAA-DB Algorithm," in *The International Joint Conference Neural Networks (IJCNN)*, 2012, pp. 1-8.

[76] S. Lavanya and K. Vishnu L. Sriram, "Imbalanced Multi Class Data Classification Using Ant Colony Optimization Algorithm," *IJAICT*, vol. 1, no. 11, pp. 877-880, March 2015.

[77] S. Alaoui and A. Lyhyaoui I. Chaïri, "Balancing Distribution of Intrusion Detection Data Using Sample Selection," *Journal of Information Security Research*, vol. 3, no. 4, pp. 153-, December 2012.

[78] W. Prachuabsupakij and N. Soonthornphisaj, "A New Classification for Multiclass Imbalanced Datasets Based on Clustering Approach," in *The 26th Annual Conference of the Japanese Society for Artificial Intelligence*, 2012.

[79] H. Leung and K. Trpkov S. Chandana, "Staging of Prostate Cancer Using Automatic Feature Selection, Sampling and Dempster-Shafer Fusion," *Cancer Informatics*, vol. 7, pp. 57-73, 2009.

[80] S. B. Rafiah and N. Rao K. Suvarnavani, "Multiclass Classification for Protein Fold Prediction Using Smote," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 2, no. 11, pp. 290-296, November 2012.

[81] Y. Zhang and Y. Zhao P. Phoungphol, "Robust Multiclass Classification for Learning from Imbalanced Biomedical Data," *Tsinghua Scienve and TechnologyY*, vol. 17, no. 6, pp. 619-628, December 2012.

[82] R. M. Pote and S. P. Akarte, " Study of Multi Class Classification for imbalanced Biomedical Data," *International Journal of Application or Innovation in Engineering & Management (IJAIEM)*, vol. 3, no. 9, pp. 158-162, September 2014.

[83] F. Ahmad and R. Alfred M. S. Sainin, "Ensemble Classifier and Resampling for Imbalanced Multiclass Learning," in *Proceedings of the 5th International Conference on Computing and Informatics, ICOCI 2015*, Istanbul, Turkey, 2015, pp. 751-756.

[84] [Online]. https://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data)

[85] [Online]. https://archive.ics.uci.edu/ml/datasets/Hepatitis

[86] [Online]. https://archive.ics.uci.edu/ml/datasets/Haberman's+Survival

[87] C. Grason and C. Martin-Vide A. Abraham, "“Evolutionary Design of Intrusion Detection Programs," *International Journal of Network Security*, vol. 4, pp. 328-339, 2007.

[88] (2014, Feb) [Online]. https://archive.ics.uci.edu/ml/datasets/Glass+Identification

[89] [Online]. https://archive.ics.uci.edu/ml/datasets/Thyroid+Disease

[90] [Online]. https://archive.ics.uci.edu/ml/datasets/Statlog+(Landsat+Satellite)

[91] [Online]. https://archive.ics.uci.edu/ml/datasets/Lymphography

[92] T. M. Mitchell, *Machine Learning*. New York: MeGrew-Hill,NY, 1997.

[93] C.J.C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 955-974, 1998.

[94] Yiming Yang, Hao Wan, et al. Tie-Yan Liu, "Support Vector Machines Classification with Very Large Scale Taxonomy," *SIGKDD Explorations*, vol. 7, no. 1, pp. 36-43, 2005.

[95] • C.M. Bishop, *Neural Networks for Pattern Recognition*.: Oxford Univ. Press, 1996.

[96] S. Haykin, *Neural Networks and Learning Machines*. Prentice Hall, 2009.

[97] M. Negnevitsky, *Artificial Intelligence*, 2nd ed., 2005.

[98] M. S. Mary and V. J. Raj, "Data Classification with Neural Classifier Using Radial Basis Function with Data Reduction Using Hierarchical Clustering," *ICTAT Journal on Soft Computing*, vol. 2, 2012.

[99] A. Meroubuha, H. Ezzikouri and M. Errritali B. Hssina, "A comparative study of decision tree ID3 and C4.5," *International Journal of Advanced Computer Science and Application, Especial Issue on Advances in Vehicular Ad Hoc Networking and Applications 2014, 2014. - See more at: http*, pp. 13-19, 2014.

[100] S. Singh and P. Gupta, "Comparative Study ID3, CART and C4.5 Decision Tree Algorithm: A Survey," *International Journal of Advanced Information Science and Technology* , vol. 27, July 2014.

[101] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, pp. 5-32, 2001.

[102] G. L. McColm, "An Introduction to Random Trees," *Research on Language and Computation*, vol. 1, no. 3, pp. 203-227, September 2003.

[103] L. Breiman, "Stacked Regressions," *Machine Learning*, pp. 49-64, 1996.

[104] R. E. Schapire, "The Strength of Weak Learnability," *Machine Learning* , vol. 5, pp. 197-227, 1990.

[105] R. E. Schapire Y. Freund, "Experiments with a new boosting algorithm," in *Thirteenth International Conference on Machine Learning*, San Francisco, 1996, pp. 148-156.

[106] Y. Yin, C. Dong, G. Yang and G. Zhou X. Gue, "On the Class Imbalance Problem," in *Fourth International Conference on Natural Computation, IEEE*, 2008.

[107] T. G. Dietterich and Ghulum Bakiri, "Solving Multiclass Learning Problems via Error Correcting Output Codes," *Journal of Artifical Intelligence Research* , vol. 2, pp. 263-286, 1995.

[108] G. A. Montazer and S. Escalera M. A. Bagheri, "Error Correcting Output Codes for multiclass classification:Application to two image vision problems," in *he 16th CSI International Symposium on Artificial Intelligence and Signal Processing* , 2012, pp. 508-513.