Cluster-Based Under-Sampling with Random Forest for Multi-Class Imbalanced Classification

Md. Yasir Arafat

Department of Computer Science and Engineering

United International University

A thesis submitted for the degree of

MSc in Computer Science & Engineering

January 2018

Declaration

I, Md.Yasir Arafat, declare that this thesis titled, "Cluster-Based Under-Sampling with Random Forest for Multi-Class Imbalanced Classification" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a MSc degree at United International University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at United International University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

Md. Yasir Arafat

Certificate

I do hereby declare that the research works embodied in this thesis "Cluster-Based Under-Sampling with Random Forest for Multi-Class Imbalanced Classification". is the outcome of an original work carried out by Mr. Md. Yasir Arafat under my supervision.

I further certify that the dissertation meets the requirements and the standard for the degree of MSc in Computer Science and Engineering.

Signed: _____

Date: _____

Dr. Dewan Md. Farid, Supervisor,

Associate Professor,
Department of Computer Science and Engineering,
United International University,
Dhaka - 1209, Bangladesh.

Abstract

Multi-class imbalanced classification has emerged as a very challenging research area in machine learning for data mining applications. It occurs when the number of training instances representing majority class instances is much higher than that of minority class instances. Existing machine learning algorithms provide a good accuracy when classifying majority class instances, but ignore misclassify the minority class instances. However, the minority class instances hold the most vital information and misclassifying them can lead to serious problems. Several sampling techniques with ensemble learning have been proposed for binary-class imbalanced classification in the last decade. In this work, we propose a new ensemble learning technique by employing cluster-based under-sampling with random forest algorithm for dealing with multiclass highly imbalanced data classification. The proposed approach cluster the majority class instances and then select the most informative majority class instances in each cluster to form several balanced datasets. After that random forest algorithm is applied on balanced datasets and applied majority voting technique to classify test/ new instances. We tested the performance of our proposed method with existing popular sampling with boosting methods like: AdaBoost, RUSBoost, and SMOTEBoost on 13 benchmark imbalanced datasets. The experimental results show that the proposed cluster-based under-sampling with random forest technique achieved high accuracy for classifying both majority and minority class instances in compare with existing Methods.

I would like to dedicate this thesis to my honourable teachers, parents, children and last but not least my wife, whose support and inspiration fortify to achieve the goal.

Published Papers

Work relating to the research presented in this thesis has been published by the author in the following conferences:

1. Md Yasir Arafat, Sabera Hoque, and Dewan Md. Farid, "Cluster-based Under-sampling with Random Forest for Multi-Class Imbalanced Classification", 11th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), December 6-8, 2017, Colombo, Sri Lanka, pp. 1-6.
2. Sabera Hoque, Md. Yasir Arafat, and Dewan Md. Farid, "Machine Learning for Mining Imbalanced Data, International Conference on Emerging Technology in Data Mining and Information Security (IEMIS 2018)", February 23-25, 2018, Kolkata, India (Accepted).

Acknowledgements

I would like to start by expressing my deepest gratitude to the Almighty Allah for giving me the ability and the strength to fulfil the task successfully within the scheduled time.

The thesis title "Cluster-Based Under-Sampling with Random Forest for Multi-Class Imbalanced Classification" has been prepared to fulfil the requirement of MCSE degree. I am very much fortunate that l have received sincere guidance, supervision, and cooperation from various persons.

I would like to express my heartiest gratitude to my supervisor, Dr. Dewan Md. Farid, Associate Professor, United International University, for his continuous guidance, encouragement and patience, and for giving me the opportunity to do this work. His valuable suggestions and strict guidance made it possible to prepare a well-organised thesis report.

I am also grateful to Dr. Chowdhury Mofizur Rahman(Professor and Vice Chancellor (In-Charge), Interim Dean), Dr. Mohammad Nurul Huda (Professor and Coordinator - MSCSE), Dr. Hasan Sarwar (Professor), Dr Salekul Islam (Associate Professor and Head of the Dept.), Dr. Khondoker Abdullah Al Mamun(Associate Professor and Director - AIMS Lab), Dr. Swakkhar Shatabda(Associate Professor and Undergraduate Program Coordinator) and some other faculty members who effortlessly working to enlightening our Computer Science and Engineering department.

I must thank our external reviewer Professor Dr. Sayed Akhter Hussain for his valuable time on my thesis work.

Moreover, my deepest gratitude and love to my parents for their support, encouragement and endless love.

Furthermore, I would like to thank my better half who was the partner throughout my master's degree.

Finally, Last but not least my 4 years old son, Your maturity is way beyond your age. You could understand that I was busy with my job and education. Your silence consideration and support was praisable. As a son, you are truly special. In our life's horizon, you are the brightest star. Thanks for being the wonderful son that you are.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Supervised learning, which is also known as classification and regression, is a challenging research area in data mining and machine learning. It builds a classification model from training data and classifies the test or new data using the model [1, 2]. As real-world data sets are mostly high dimensional, multi-class and highly imbalanced, which cause learning algorithms doomed to achieve high classification accuracy. To resolves this complication a significant number of ensemble classifier with sampling methods for classifying multi class imbalanced data have been proposed in the last decade [3–5]. Ensemble classifier enhances the performance of individual classifiers by combining various machine-learning algorithms and conjoins multiple hypotheses to form an advance composite model [2]. The sampling technique uses either under-sampling of the majority class instances or over-sampling the minority class instances in order to beget a more balanced data distribution. However, both under-sampling and over-sampling techniques have some deficiency like the random under-sampling technique may loss some potentially useful informative training instances. On the other hand, over-sampling with replacement caused ineffectiveness in increasing the minority class instances as it either increases the likelihood of overfitting or demonstrated of redundancy [6].

In real-world applications, the majority class instances dominate the minority class instances, but the minority class instances representing the information of greater concern as compared to the majority class instances [7]. The traditional machine learning and data mining algorithms like decision tree (DT) [1, 2], k-nearest neighbour (kNN) [1], and naïve Bayes (NB) classifier [8] build the classification models to maximise the clas-

sification accuracy by classifying majority class instances but ignore the minority class instances. So, for dealing with class imbalance problems the most approved practices are sampling techniques, ensemble classifier methods and cost-sensitive learning methods. The sampling techniques (under-sampling and over-sampling) either remove the majority class instances from the imbalanced data or add the minority class instances into the imbalanced data to get the balanced data. Bagging and Boosting that is well known popular ensemble method also adopt sampling technique in its each iteration for classifying imbalanced data. The cost sensitive learning technique assigns different costs in relation to the misclassification error of classes. Generally high cost is assigned for the minority class instances and low cost accredit for the majority class. However, the classification results are not stable in cost-sensitive learning methods as it is very challenging to achieve a precise misclassification cost. Furthermore, different misclassification costs can potentially lead to different conclusions. The major approaches for resolving class imbalance problems can be split into two groups: (1) external approach also known as data balancing methods, which preprocess the high imbalanced data to obtain the balanced data, and (2) internal approach, which reshape existing learning algorithms in order to decline their sensibility to the class imbalance when learning from the imbalanced data.

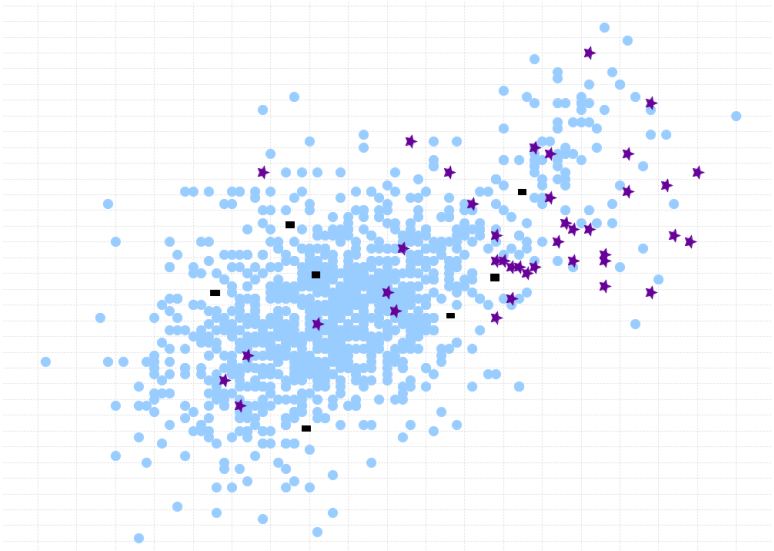

Figure 1.1: An example of imbalance data distribution.

1.1 Motivation

Class imbalance problem is very common in the real world and continuous imbalance data are generating by machines, social media and human. This problem occurs when the ratio between majority class instances and minority class instances is too high. Class imbalance problem can be found very often in various sectors like financial fraud detection, diseases diagnosis, face detection etc.

Traditional machine learning methods work fine when the number of majority and minority class instances is almost equal. Most of the cases the outlier or noble class are ignored. It is very important to find the fraud detection in ATM or an e-commerce company. E-commerce company like Amazon, Netflix spent a lot of money to catch fraudulent transactions. So it is very important to find meaningful information from these kind of imbalance data.

It is very important to know on how to handle the imbalance data in supervised learning. We can mitigate by using data balancing techniques like sampling based approaches, cost function based approaches or ensemble methods. As Machine learning becomes a great challenge in the present world as for instances Google car, fraud detection, online recommendation engines, which inspires us to work on multi-class imbalanced data classification.

1.2 Objectives of the Thesis

To design and develop an optimal ensemble classifier for classifying imbalanced data. The main objective is to find the most informative majority class instances to form several balanced datasets and apply the random forest algorithm on every balanced data-sets to build an ensemble classification model. Random forest is a strong and most popular ensemble learning technique that uses decision tree induction algorithm and majority voting technique to classify new or unseen test instances. Random forest (RF) algorithm is an ensemble classifier that uses several decision trees with majority voting techniques for classifying test instances.

To find the most informative majority class instances we used clustering technique. Clustering technique groups the majority class instances into several clusters, and we select the informative majority class instances from each cluster. To select the informative instances we consider the instances that are close to the center of the cluster

and border of the cluster. The cluster-based under-sampling technique helps us to avoid losing the informative instances. The objective of this thesis is to find the most informative majority class instances

1.3 Thesis Contributions

We have proposed a new technique, a clustering-based under-sampling approach with random forest classifier for multi-class imbalanced data classification. Firstly, the imbalanced data is divided into two sub-data sets: majority class instances and minority class instances. Then majority class instances are clustered into several clusters and equal number of informative majority class instances with compare to minority class instances are selected from each cluster to form balanced data sets. Clustering groups the instances in the data space according to their similarity and ensures that each cluster contains instances more homologous to each other than to those of other clusters. Secondly, the random forest classifier is applied on balanced data sets to build a composite classification model. Finally, the random forest classifier uses majority voting to classify the unknown or test instances. The basic idea of this thesis is to select informative majority class instances instead of randomly removing the majority class instances to form balanced data. Therefore, the proposed approach combines the sampling and random forest methods to form a proficient, feasible and an efficacious algorithm for class imbalance learning. The performance of proposed method is tested and compared with AdaBoost, RUSBoost, and SMOTEBoost algorithms on 13 imbalance benchmark datasets. Based on the experimental results, we can endorse that combining clustering-based under-sampling approach with random forest algorithm is a promising technique for improving the bias of class imbalance problems.

I have done the following work-out to achieve better performance of my proposed model.

- Emperical research on different imbalance data with different technique.
- Implement new different feasible techniques to classify multi-class imbalanced data.
- Implement our proposed method with python language include scikit-learn library.

1.4 Organization of the Thesis

The rest of this book is organized as follows.

Chapter 2 Presents the data balancing methods.

Chapter 3 Presents the proposed algorithm for imbalance data classification.

Chapter 4 Discusses the results and experimental analysis.

Chapter 5 Presents the conclusions and the future works.

Chapter 2

Related Work

Several research have been done employing sampling techniques, ensemble methods, and cost-sensitive learning methods to handle binary-class imbalanced classification problems in the last decade [9, 10]. This chapter presents the most existing method for imbalanced data classification.

2.1 Sampling Methods

To resolve class imbalance problem sampling methods are divided into two parts. Such as - (1) Under-sampling, (2) Over-sampling. Random under-sampling and random oversampling are non-heuristic methods that were originally composed of this theory as fundamental level. This non-heuristic approach aspired to balance class distribution through the random replication of minority class examples.

2.1.1 Over-sampling Methods:

SMOTE: In over-sampling methods minority classes are increased by adding instances. Chawla et al. [11] introduced an over-sampling technique called SMOTE (Synthetic Minority Over-sampling Technique) algorithm where over-sampling is conducted on the minority class instances through the formation of synthetic minority class instances instead of over-sampling with reinstatement. SMOTE beget synthetic instances by performing in feature space rather than data space employing k minority class nearest neighbours. In some experimental result showed that the combination of over-sampling

with under-sampling performed better. It is done by interpolation of minority class instances, which congregate with nearest one. SMOTE generated samples could be almost identical to the existing minority class instances that created redundancy. Santos et al. [12] implemented a cluster-based (k-means) over-sampling approach where reduced sized clusters oversampled by adopting SMOTE. This approach examined merging the minority class instances from the multiple over-sampled data sets. Blagus and Lusa [13] investigated the behaviour of SMOTE on highly dimensional imbalanced data sets and found that feature selection is necessary for SMOTE with k-nearest neighbour (kNN), as SMOTE strongly biases the classifier towards the minority class.

MSMOTE: The variant of Synthetic Minority Over-sampling Technique (SMOTE) is Modified Synthetic Minority Over-sampling Technique (MSMOTE), which additionally considers the distribution of minor class instances. Since the imbalance ratio of the dataset is not elevated, the under-sampling method usually works better than over-sampling methods. MSMOTE method improving the classification performance when training data is imbalanced [14].

2.1.2 Unser-sampling Methods:

In under-sampling methods, a number of majority class instances are randomly eliminated from imbalanced data set to create a balanced data set [9]. This kind of elimination may loss the informative training instances from the majority class especially if the number of instances of the majority class is larger than minority class instances (for example majority to minority ratio is 100:1).

RUS-I: Batista et al. [15] proposed the random under-sampling technique, where the core objective of the model is not only to balanced the training data but also to eliminate noisy instances. Though elimination of noisy instances might help in discovering well-defined clusters, this method can discard the potentially useful data that could be significant for the induction procedure.

2.2 Cost Sensitive Measure

This method deal with the class imbalance problem by accrediting more misclassification cost to minority class instances for the underlying classifier. The misclassification cost can be used in the cost function and the classifier can optimize the cost function.

The classifier will now treat both majority and minority classes equally due to their underlying costs. The greatest challenge in this method is assigning the costs as they are difficult to be evolved from datasets.

Cost-sensitive Tree: Cost-sensitive tree induction via instance weighting was proposed in a paper written by Kai Ming Ting [16]. The idea of information gain is used to construct a tree. The attribute with the highest entropy is assigned as the root, the one with lower gain becomes the child of the node and so on. After the tree is constructed, pruning is carried out to reduce the size of the tree. This results in a tree with least error and therefore this method has a higher accuracy.

Cost-sensitive Neural Network: Zhou and Liu [17] presented an approach that involves the cost-sensitive neural network. Sampling, threshold moving, hard ensemble and soft ensemble were all considered while training the cost-sensitive neural network. Overall, threshold-moving and soft ensembles were figured out to be a good choice for training cost sensitive neural networks. However, this method became more difficult with the increase in the degree of class imbalance.

2.3 Ensemble Methods

Ensemble learning methods is a congregational approach established by merging multiple base learners, which may either same or different [18–20]. This approach usually helps to increase the prediction capability of the individual base classifiers, thus make it adaptive to different datasets. This method has been finding very useful in a wide variety of real-life problems.

Sun et al. [6] proposed an ensemble technique in order to deal with binary class imbalance problems. In this approach, instances of the majority class were split into several groups/ sub-data sets, where each subset has a similar number of minority class instances. Thus, a number of balanced datasets were generated. Then, each balanced data set was bestowed in creating a binary classifier. Lastly, these binary classifiers were combined in creating an ensemble model to classifier the new data.

Galar et al. [21] proposed an ensemble algorithm by Evolutionary Under Sampling (EUS) approach, named EUSBoost, to classify highly imbalanced data sets. EUS engendered distinct sub-datasets adopting random under-sampling technique and obtained an outstanding under-sampled data set of the authentic data set that cannot

be remodelled any more. SMOTEBoost, RUSBoost and EUSBoost applied data sampling techniques into the AdaBoost algorithm by considering both the minority class instances and the majority class instances. Yen and Lee [22] presented a cluster-based under-sampling method by clustering all the training instances (minority class instances and majority class instances) into some clusters. A convenient sum of majority class instances is selected from each cluster by this approach, based on the ratio of majority class instances to the minority class instances in the particular cluster.

2.3.1 Bagging

In this section hybrid based bagging methods related work will be concentrated.

UnderBagging: The core idea of UnderBagging method is to educate each of the single components of the ensemble with a balanced learning instances illustrated by Barandela et al. [23]. By replacing an individual classification model, (Here the 1-NN rule) with an imbalanced training set, by an amalgamation of various classifiers, each using a balanced training set for its learning process. To earn this, as many training sub-samples as possible to get balanced subsets are beget. The number of sub-samples will be discovered by the difference between the number of prototypes from the majority class and that of the minority class. This workflow makes it possible to properly handle the difficulties of the imbalance, and stay away from the implicit disadvantages to both the over and undersampling techniques.

OverBagging: According to Wang et al., OverBagging forms each subset simply by over-sampling minority classes randomly [24] . After formation, a majority vote is accomplished every time a new instance arrives. After that, each classifier will give their decision and final classification decision follows the majority voted class. If a tie appears, then the class with minor instances is returned. From training phase to testing phase we can be illustrated the whole process in 3 steps - re-sampling, constructing ensemble and voting. As there may be multiple minority and majority classes, it is tough to take a decision which re-sampling rate we should use. The output of the algorithm shows that diversity dominates recall value notably. Basically, larger diversity causes better recall for a minority but worse recall for majority classes. As diversity decreases, recall values tend to be smaller for minority classes. This is because diversity

enhances the probability of classifying an instance as the minority when accuracy is not high enough [24].

UnderOverBagging: OverBagging is utterly similar to UnderBagging either using random oversampling or random undersampling. So according to this concept UnderOverBagging trains the base classifier with varying resampling rates using random oversampling and random undersampling according to necessity [24].

SMOTEBagging: SMOTEBagging is differed from UnderBagging and OverBagging presented by Wang et al. [24], when involving in subset creation of synthetic instances. As claimed by the SMOTE algorithm, first of all, two parameters need to be fixed, which are k nearest neighbors and the amount of over-sampling from minority class N. Here N=100, 200, 300, 400 and 500 i.e five nearest neighbour. The relative class must be considered when distribution among all minority classes after resampling instead of over-sampling each class independently by using different N values. SMOTEBagging uses SMOTE as a preprocessing technique for the base classifier. It shows improved predictive accuracy in imbalance classification problem [24].

2.3.2 Boosting

According to Freund et al. [25], AdaBoost is one of the best boosting technique that can produce highly accurate prediction rule by merging many weak and inaccurate rules. Each classifier trained serially with the goal of correctly classifying instances in every iteration that were incorrectly classified in the previous classification. Seiffert et al. [26] designed a different hybrid sampling/boosting algorithm, called RUSBoost, which put together Random Under-Sampling (RUS) with AdaBoost algorithm. RUS randomly removes the majority class instances to construct a balanced data. RUSBoost was built based on the SMOTEBoost (synthetic minority over-sampling with AdaBoost) algorithm [27]. SMOTEBoost was completed upon over-sampling approach with AdaBoost algorithm.

RUSBoost: RUSBoost is a novel hybrid sampling/boosting algorithm presented by Hulse [28], which is used for learning from skewed training data and designed to increase the performance of models trained on skewed data. This is an easier and swifter option to SMOTEBoost which is a combination of boosting and data sampling technique. Both RUSBoost and SMOTEBoost acquaint data sampling into the AdaBoost algorithm. SMOTEBoost does so use an oversampling technique (synthetic minority over-sampling

technique (SMOTE) [11] which creates novel minority class instances by extrapolating among existing instances. Whereas, RUSBoost implements RUS, which is a method that randomly eliminates instances from the majority class. RUS has been conferred to perform great despite its simplicity [28]. The ideas for carrying RUS into the boosting process are simplicity, speed, and performance. SMOTE is a more difficult and time-consuming data sampling system than RUS.

Moreover, SMOTE is an oversampling technique which increased model training time. SMOTEBoost expands this disadvantage since boosting needs the training of an ensemble of models. On the other side, RUS decreases the time needed to build a model, especially when creating an ensemble of models, which is the case in boosting.

One of the most common data sampling techniques (largely due to its simplicity) is RUS. Unlike more complex data sampling algorithms, RUS makes no attempt to "intelligently" remove examples from the training data. Instead, RUS simply removes examples from the majority class at random until a desired class distribution is achieved.

SMOTEBoost: One of the most popular boosting techniques is SMOTEBoost proposed by [11] which consolidates an intellectual oversampling technique (SMOTE) with AdaBoost, occurring in a highly efficient hybrid approach to learning from imbalanced data.

SMOTEBoost, which was introduced by Chawla et al. [11], consolidates the SMOTE algorithm with Ada-Boost, finishing in a hybrid sampling/boosting algorithm that outperforms both SMOTE and AdaBoost. Similar to RUSBoost, SMOTEBoost is based on the AdaBoost.M2 algorithm. During each round of boosting the weak hypothesis building earlier, SMOTE is applied to the training data to produce a more balanced training data set.

Hence, the algorithm for SMOTEBoost is much alike to that of RUSBoost, The purpose of SMOTE at this point has two impediments that RUSBoost is composed to overcome. First, it enhances the complexity of the algorithm. SMOTE must attain the k nearest neighbors of the minority class instances and extrapolate between them to make new instances. On the other hand, RUS simply erases the majority class instances randomly. Next, since SMOTE is an oversampling procedure, it appears in extended model training times. This effect is intensified by SMOTEBoost's use of boosting since many models must be developed employing larger training data sets.

On the other hand, RUS results in less training data sets and, therefore, it requires less model training times.

2.4 Summary

Several research works have been done employing sampling techniques, ensemble methods, and cost-sensitive learning methods to handle imbalanced binary classification problems in the last decade [9, 10]. Still, a lot of research work has been doing by the researcher to find insight from imbalanced data for multi-class classification problem.

Chapter 3

Proposed Data Balancing Method

In this chapter we will discuss on different data balancing techniques and introduce our proposed methods.

3.1 Data Balancing Methods

There are different types of data balancing methods such as under-sampling, oversampling, cost-sensative learning and ensemble learning. In this section We will give a basic idea on these balancing methods.

3.2 Sampling Methods

There are two different sampling methods i) Under-sampling ii) Over-sampling.

3.2.1 Under-sampling Methods

In under-sampling methods, a number of majority class instances are randomly eliminated from imbalanced data set to create a balanced data set.

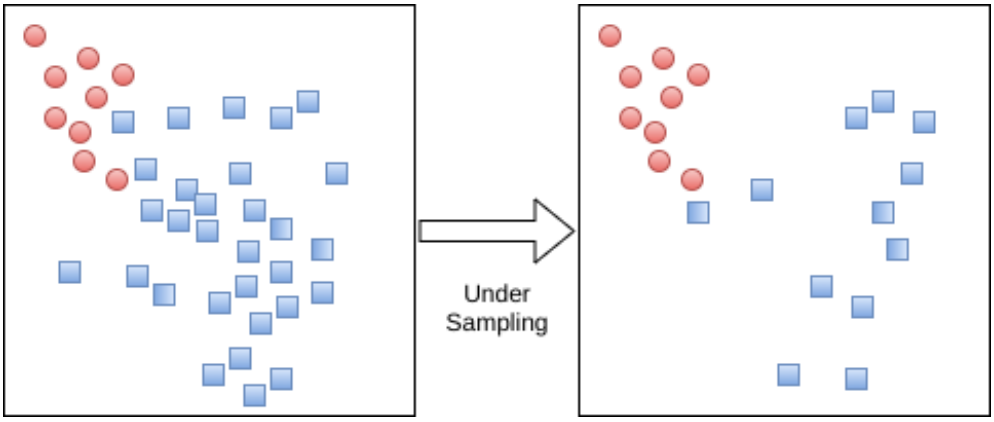

Figure 3.1: Under Sampling methods remove majority instances.

This kind of elimination may loss the informative training instances from the majority class especially if the number of instances of the majority class is larger than minority class instances (for example majority to minority ratio is 100:1). Figure 3.2 shows the random under-sampling (RUS) approach in sampling technique for creating balanced data set.

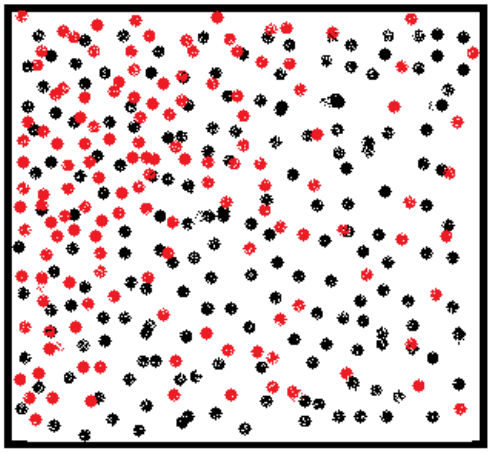

Figure 3.2: Random under-sampling (RUS) where black and red dots are representing the majority and minority class instances respectively in imbalanced data set. The red dots are selected randomly to form a balanced data set.

3.2.2 Over-sampling Methods

In over-sampling method, the increment of the minority class instances can be done in different ways. Random over-sampling is the most common form among them. Here minority class instances are clone to ensure both class instances are balanced. This process creates a high probability of over-fitting due to the same instances may clone over and over. The popular over-sampling technique is SMOTE (Synthetic Minority class Oversampling Technique). This method creates synthetic instances of the minority class instead of duplication [11]. It is done by interpolation of minority class instances, which congregate with nearest one. SMOTE generated samples could be almost identical to the existing minority class instances that created redundancy. The variant of Synthetic Minority Over-sampling Technique (SMOTE) is Modified Synthetic Minority Over-sampling Technique (MSMOTE), which additionally considers the distribution of minor class instances. This method functions by splitting the data belonging to the minority class into three categories (border, safe & latent noise) based on their distance with all training instances. Since the imbalance ratio of the dataset is not elevated, the under-sampling method usually works better than over-sampling methods [7].

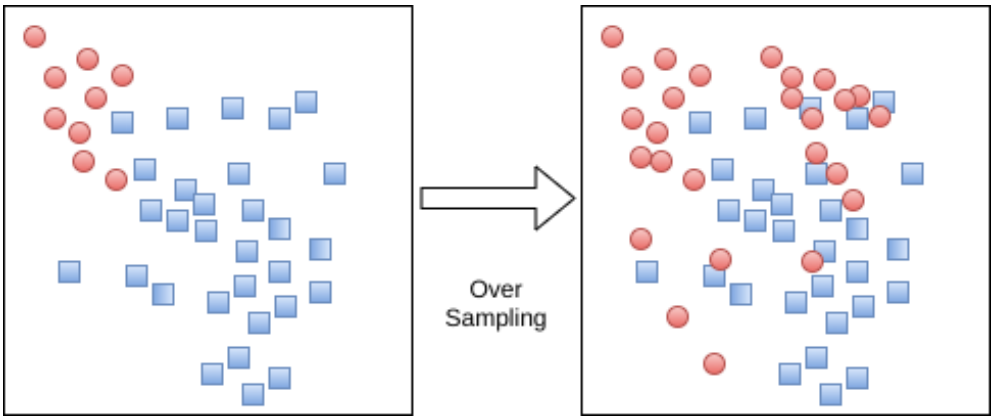

Figure 3.3: Over Sampling add minority instances.

3.2.3 Cost-sensitive learning Methods

This method deal with the class imbalance problem by accrediting more misclassification cost to minority class instances for the underlying classifier. The misclassification cost can be used in the cost function and the classifier can optimize the cost function.

The classifier will now treat both majority and minority classes equally due to their underlying costs. The greatest challenge in this method is assigning the costs to majority and minority classes as they are difficult to be evolved from datasets.

3.2.4 Ensemble Learning Methods

Ensemble learning methods is a congregational approach established by merging multiple base learners, which may either same or different [18–20]. This approach usually helps to increase the prediction capability of the individual base classifiers, thus make it adaptive to different data sets. This method has been finding very useful in a wide variety of real-life problems.

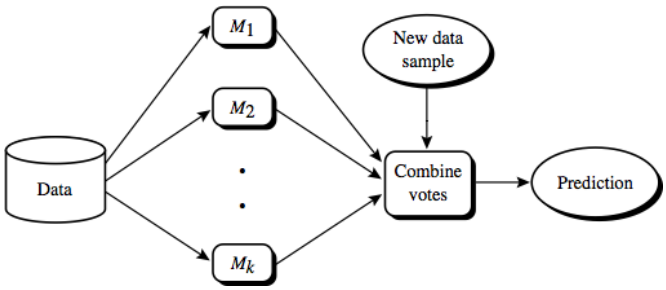

Figure 3.4: Ensemble Model.

3.2.5 Random Forest

In this method attribute bagging technique is applied for creating sub-data sets from original data set. Attribute bagging also an ensemble learning approach that select attributes randomly from the full space. A random forest is a collection of decision trees where identically disbursed random vectors and each tree cast a vote for the most popular class of input instance. Trees are one of the most favoured and accurate learning methods commonly used for data exploration. For many data sets, it fabricates exact classifier and executed smoothly on extensive databases.

Without variable deletion this classifier able to handle enormous input variables. Apart from that, notable enhancement in classification accuracy has resulted from developing an ensemble of trees and allows them to vote for the most popular class. However, with noisy classification/regression tasks, random forests have been detected as overfitted.

3.2.6 Bagging

Bagging trains multiple base learners through sampling with replacement or without replacement in parallel. The base learners while considering each instance with equal weights use the subsets. The main target of bagging is to reduce the variance of some high variance low bias base classifiers while retaining the low bias. Average the output of the base learners does this. The output of the individual base learners is considered as votes to determine the final prediction. However, bagging itself cannot tackle imbalanced classification problems. In order to make bagging compatible in this domain, sampling methods like under-sampling and over-sampling have been applied in each iteration before training the base learners.

3.2.7 Boosting

Boosting is similar to bagging, it combines multiple base learners to obtain a result based on voting technique but trains the models in a sequential manner [26, 29]. Boosting is vitiated by it assigns weight to instances according to how hard they are to classify, thus setting a high weight to instances that are misclassified. Weights are assigned to the base learners as well according to their predictive accuracy and this is taken into consideration when classifying a new test instance. If the training instances are misclassified then the weights of these instances are increased. On the other side, the weights of the correctly classified instances are decreased. Therefore, each base classifier is trained with a different distribution of the training data that makes it diverse. The base learners are also assigned weights according to their performance on the training data. Boosting itself cannot deal with imbalanced datasets, so different sampling methods are applied at each iteration of boosting. This makes the training partition balanced [9, 25]. It can be somewhat referred to as a cost-sensitive method.

3.3 Proposed Method

In this section, we are presenting the proposed method that puts together clustering-based under-sampling technique with random forest algorithm. The proposed method is similar to RUSBoost and SMOTEBoost methods with the critical difference occurring in the sampling technique. RUSBoost uses random under-sampling on the majority class, while SMOTE-Boost uses SMOTE method to oversample the minority class

instances. In contrast, our proposed method uses cluster-based sampling from the majority class. At first, we separate the majority and minority class instances from the original dataset. Then the majority class instances are split up into different clusters and an equal number of informative majority class instances with compare to minority class instances are selected from each cluster to form several balanced data sets. The most similar instances should be grouped into one cluster, where the dissimilar instances will be in other clusters.

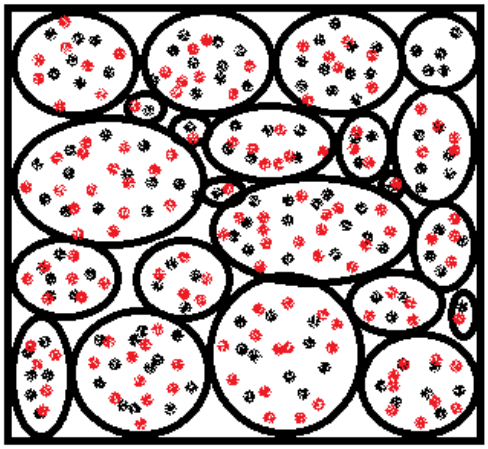

Figure 3.5: Creating several balanced data sets using cluster-based under-sampling technique, where red and black dots are representing majority and minority class instances respectively

The majority class instances are grouped into k clusters. We use ensemble clustering employing k-means and similarity based clustering to cluster the majority class instances [30]. After clustering the majority class instances, we select the informative instances from each cluster. We consider informative instances that are close to the center of the cluster and border of the cluster. Then several balanced data sets are created with informative majority class instances and minority class instances. Finally, several decision trees are generated from balanced data sets with attribute bagging technique. To classify the unknown or test instances majority voting is applied by the trees, which will produce the final prediction. The basic idea of the proposed method is to select informative majority class instances instead of randomly removing the majority class instances to form balanced data. The proposed method combines the sampling and random forest methods to form an efficient algorithm for class imbalance learning.

Figure 3 shows the proposed method for classifying imbalanced data using sampling with random forest algorithm. Algorithm presents the proposed cluster-based sampling with random forest algorithm.

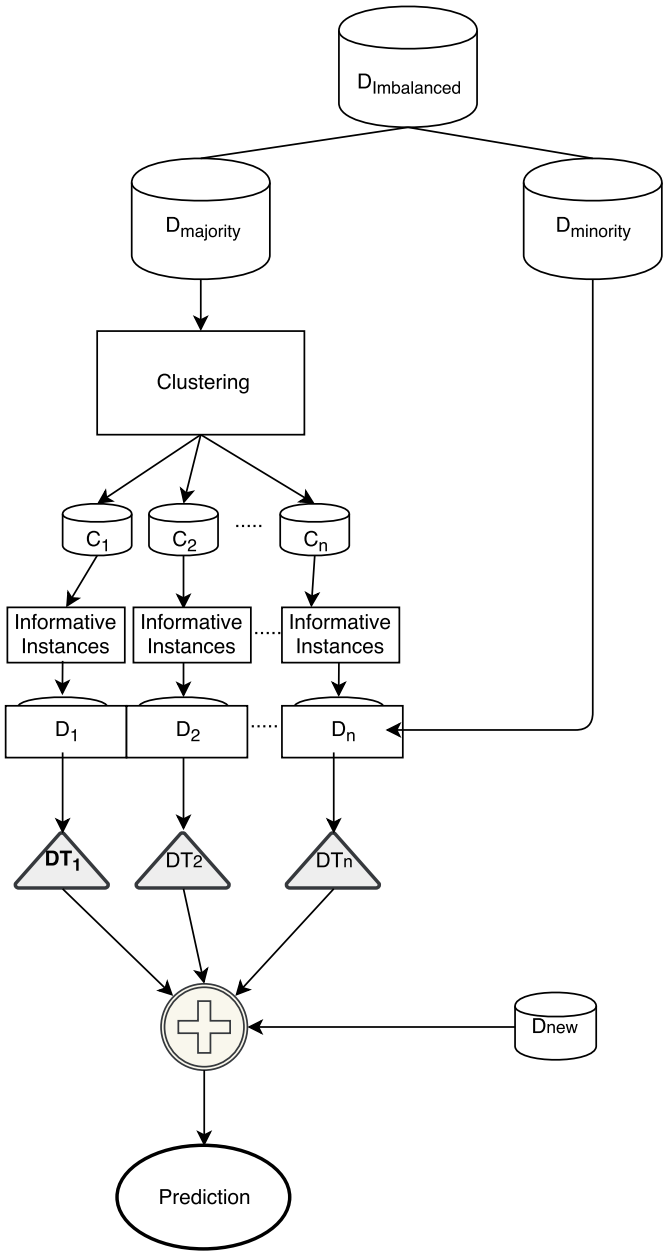

Figure 3.6: Proposed Model (Cluster based undersampling with random forest).

Algorithm 1 Cluster-based Sampling with Random Forest

Input: Training data $D_{Imbalanced}$, Ensemble clustering algorithm & Decision tree (C4.5) induction algorithm.

Output: Ensemble of trees.

Method:

1: create several balanced datasets $D_i = \{D_1, \cdots, D_n\}$ from $D_{Imbalanced}$ using cluster-based under-sampling;
2: **for** I = 1 to n **do**
3: derive a tree, DT_i from D_i employing C4.5 algorithm by randomly selected features;
4: compute the error rate of DT_i, $error(DT_i)$;
5: **if** $error(DT_i) \geq 0.5$ **then**
6: return to step 3;
7: **end if**
8: **end for**

To classify instance, x_{New} use ensemble of trees by majority voting technique;

3.4 Summary

In this thesis work, We proposed a clustering-based under-sampling approach with random forest classifier for multi-class imbalanced data classification. Firstly, the imbalanced data is divided into two sub-data sets: majority class instances and minority class instances. Then majority class instances are clustered into several clusters and equal number of in-formative majority class instances with compare to minority class instances are selected from each cluster to form balanced data sets. Clustering groups the instances in the data space according to their similarity and ensures that each cluster contains instances more homologous to each other than to those of other clusters. Secondly, the random forest classifier is applied on balanced data sets to build a composite classification model. Finally, the random forest classifier uses majority voting to classify the unknown or test instances. The basic idea of this thesis is to select informative majority class instances instead of randomly removing the majority class instances to form balanced data. Therefore, the proposed approach combines the sampling and random forest methods to form a proficient, feasible and an efficacious algorithm for class imbalance learning. The performance of proposed method is tested and compared

with AdaBoost, RUSBoost, and SMOTEBoost algorithms on 13 imbalance benchmark datasets. Based on the experimental results, we can endorse that combining clustering-based under-sampling approach with random forest algorithm is a promising technique for improv-ing the bias of class imbalance problems.

Chapter 4

Experimental Analysis

In this section, we present the experimental analysis to examine the performance of the proposed algorithm. We have used muti-class datasets with different imbalance ratio and different machine learning approaches.

4.1 Data set

We have used 13 multi-class data sets with different imbalance ratio which are taken from KEEL dataset repository 4.1. Table shows the data sets details.

Table 4.1: Imbalanced data sets description.

No.	Datasets	Instances	Features	Class Values	Imbalance Ratio
1	pima	768	8	2	1.87
2	dermatology	366	34	6	5.55
3	segment0	2308	19	2	6.02
4	led7digit	443	7	2	10.97
5	abalone9-18	731	8	2	16.4
6	yeast	1484	8	10	23.15
7	$poker - 9_v s_7$	244	10	2	29.5
8	kddcup-guess passwd_vs_satan	1642	41	2	29.98
9	yeast5	1484	8	2	38.73
10	ecoli	336	7	8	71.5
11	abalone19	4174	8	2	129.44
12	Page Blocks	548	10	5	164
13	Statlog (Shuttle)	2175	9	7	853

4.2 Performance Evaluation

4.2.1 ROC Curves

ROC curve is a graph that represents the true positive rate (TPR) along the y-axis and false positive rate (FPR) along the x-axis. The perfect classifier would have an Area Under the ROC Curve (AUC) of 1, where TPR = 1 and FPR = 0. At different cut-off points the curve shows the TPR and FPR of the classifier at different thresholds.

$$TPRate = \frac{TP}{TP + FN} \tag{4.1}$$

$$FPRate = \frac{FP}{FP + TN} \tag{4.2}$$

In the case of imbalanced classification problems, emphasis is placed particularly on the predictive accuracy of minority class instances. This would correspond to high tpr and low fpr, and thus reflected by a high AUC. We have used AUC to compare the proposed method with other state-of-the-art techniques. Following is the ROC curve for the case in hand. It means if we put the the rate of TPR and FPR in a graph that we will get from the confusion matrix will draw some curve which is known as ROC Curves.

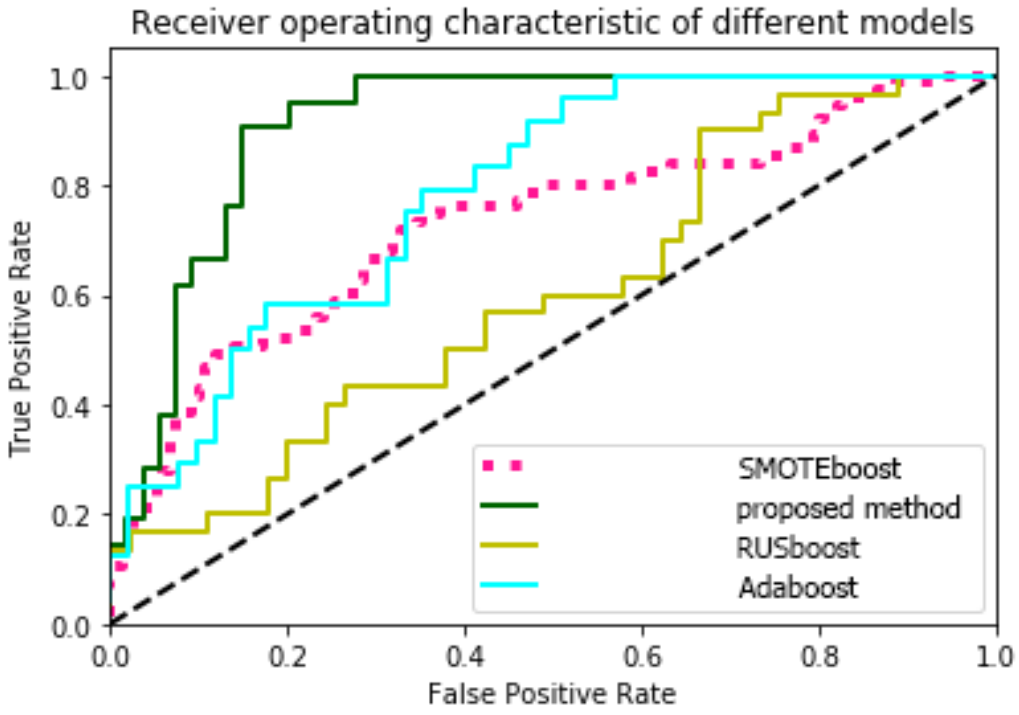

Figure 4.1: An example ROC comparison with different classification model for Yeast dataset

4.3 Results

In this section, we present the experimental analysis to examine the performance of the proposed algorithm. In our experiment, we have compared the proposed method with AdaBoost, RUSboost, and SMOTEBoost algorithms. Each data set was validated using Area Under the ROC Curve (AUC). As for the base learner, We used decision tree induction. Each of the experiments is done 5 times with 10 fold cross validation and their mean scores are shown in Table 4.2.

From the experimental results it is clear that, when the imbalance ratio is within a distinct range the proposed algorithm performs outstanding almost all time. As the imbalance ratio getting higher proposed algorithm starts to outperform rather than all the other methods significantly. As because the proposed algorithm does not focused on making the ratio of majority and minority class examples 1:1. So for this the sub-sampled training dataset holds a better representation of the majority class

Table 4.2: Average performance of the AdaBoost, RUSBoost, SMOTEBoost, and Proposed method on 13 imbalanced datasets.

Datasets	AdaBoost	RUSBoost	SOMTEBoost	Proposed
pima	0.6223	0.6376	0.6597	**0.671**
dermatology	**0.9342**	0.43	0.632	0.55
segment0	**0.996**	0.957	0.951	0.951
led7digit	0.8937	0.8279	0.9373	**0.947**
abalone9-18	0.6934	0.7051	0.7195	**0.731**
yeast	0.7589	0.7382	0.741	**0.769**
$poker - 9_v s_7$	0.642	0.589	0.6708	**0.967**
kddcup-guess passwd_vs_satan	0.8324	0.8681	0.8513	**0.879**
yeast5	0.9231	0.887	**0.9253**	0.9
ecoli	0.6354	0.517	**0.6597**	0.657
abalone19	0.5723	0.5923	0.5583	**0.610**
Page Blocks	0.7605	0.5975	0.8123	**0.899**
Statlog (Shuttle)	0.8331	0.727	0.8445	**0.89**

while remaining imbalance itself. RUSBoost shows a performance with high variance especially with highly imbalanced datasets, thus it shows a poor performance if the mean results are chosen. But, if the best results are chosen from 10 experiments then RUSBoost outperforms the other methods including the proposed method on the majority of the datasets as shown in Table 4.3. The best result of proposed method is usually quite close to the average result thus proving low variance in its performance.

Table 4.3: The best result using AdaBoost, RUSBoost, SMOTEBoost, and Proposed method in each dataset is stressed in bold-face.

Datasets	AdaBoost	RUSBoost	SOMTEBoost	Proposed
pima	0.6653	**0.6796**	0.667	0.6679
dermatology	**0.9**	0.6935	0.632	0.54
segment0	**0.998**	0.9805	0.9642	0.9523
led7digit	0.9157	0.886	0.9485	**0.9544**
abalone9-18	0.707	**0.805**	0.7029	0.7434
yeast	0.7839	**0.801**	0.7653	0.7895
$poker - 9_v s_7$	0.657	0.7395	0.69	**0.9685**
kddcup-guess passwd_vs_satan	0.8578	**0.9123**	0.8602	0.8849
yeast5	0.9324	**0.9515**	0.9391	0.9126
ecoli	0.6455	**0.677**	0.6753	0.6656
abalone19	0.5796	**0.6498**	0.5876	0.6176
Page Blocks	0.78	**0.937**	0.8592	0.9007
Statlog (Shuttle)	0.8567	**0.8989**	0.8516	0.8967

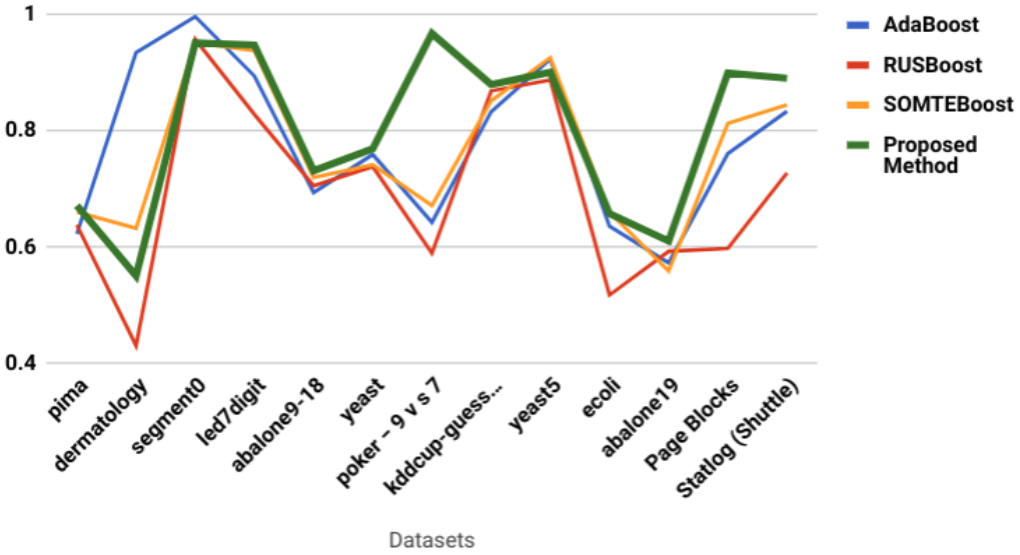

Figure 4.2: Performance comparison of AdaBoost, RUSBoost, SMOTEBoost, and Proposed method

4.4 Summary

In this chapter, we have tried to focus on our experiment and our result comparison. For our experiments, the datasets are collected from Keel and UCI Machine Learning Repositories. For the development of the model, we have used python Scikit learn where NumPy is the fundamental package for scientific computing, Matplotlib for data plotting, Panda for data structures and data analysis.

After developing the model with decision tree induction as a base learner, each dataset experiments 5 times with 10 fold cross-validation and calculate their mean. Each result validated using Area Under the ROC Curve(AUC)

Chapter 5

Conclusions and Future Work

5.1 Conclusions

Most of the machine learning for data mining algorithms usually focus on majority class instances and neglect the minority class instances. To construct an effective classifier, which can correctly classify the instances of the minority class is a really very challenging task. Now-a-days, computational intelligence researchers have been using several sampling with ensemble classifiers for dealing with class imbalance problems. This work introduced a new approach for multi-class imbalanced classification using cluster-based under-sampling with random forest algorithm. We used cluster based under-sampling technique for selecting the informative majority class instances. We considered informative instances that are close to center of the cluster and border of the cluster. Then several balanced data sets are created by informative majority class instances with minority class instances. We used random forest classifier for classifying new data, which is most popular ensemble classification technique. The performance of proposed algorithm has been compared with the most competent boosting techniques like AdaBoost, RUSBoost, and SMOTEBoost algorithms. From the experimental results, we have found that the proposed method performed impeccably when compared to these popular techniques on data sets having high-class imbalance ratio.

5.2 Future Work

In future, We would like to conduct further experiments to investigate the performance of the cluster-based under-sampling with random forest algorithm in high-dimensional

big data and data streaming environments. Also, we will find the informative test instances from the test data that will help us to validate the ensemble classifiers for mining imbalanced data.

Bibliography

[1] D. M. Farid, M. A. Al-Mamun, B. Manderick, and A. Nowe, "An adaptive rule-based classifier for mining big biological data," *Expert Systems with Applications*, vol. 64, pp. 305–316, December 2016.

[2] D. M. Farid, L. Zhang, A. Hossain, C. M. Rahman, R. Strachan, G. Sexton, and K. Dahal, "An adaptive ensemble classifier for mining concept drifting data streams," *Expert Systems with Applications*, vol. 40, no. 15, pp. 5895–5906, November 2013.

[3] H. He and E. A. Garcia, "Learning from Imbalanced Data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, June 2009.

[4] Y. Sun, A. K. C. Wong, and M. S. Kamel, "Classification of imbalance data: A review," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 4, pp. 687–719, June 2009.

[5] M. Wasikowski and X.-w. Chen, "Combating the Small Sample Class Imbalance Problem Using Feature Selection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1388–1400, October 2009.

[6] Z. Sun, Q. Song, X. Zhu, H. Sun, B. Xu, and Y. Zhou, "A novel ensemble method for classifying imbalanced data," *Pattern Recognition*, vol. 48, no. 5, pp. 1623–1637, May 2015.

[7] D. M. Farid, A. Nowé, and B. Manderick, "A New Data Balancing Method for Classifying Multi-Class Imbalanced Genomic Data," *25th Belgian-Dutch Conference on Machine Learning (Benelearn)*, pp. 1–2, 12-13 September 2016.

[8] D. M. Farid, L. Zhang, C. M. Rahman, M. Hossain, and R. Strachan, "Hybrid decision tree and naïve bayes classifiers for multi-class classification tasks," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1937–1946, March 2014.

[9] C. Beyan and R. Fisher, "Classifying imbalanced data sets using similarity based hierarchical decomposition," *Pattern Recognition*, vol. 48, no. 5, pp. 1653–1672, May 2015.

[10] A. D. Pozzolo, O. Caelen, and G. Bontempi, "When is undersampling effective in unbalanced classification tasks?" in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2015, pp. 200–215.

[11] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, June 2002.

[12] M. S. Santos, P. H. Abreu, P. J. García-Laencina, A. Simão, and A. Carvalho, "A new cluster-based oversampling method for improving survival prediction of hepatocellular carcinoma patients," *Journal of Biomedical Informatics*, vol. 58, pp. 49–59, September 2015.

[13] R. Blagus and L. Lusa, "SMOTE for high-dimensional class-imbalanced data," *BMC Bioinformatics*, vol. 14, no. 106, pp. 1–16, March 2013.

[14] S. Hu, Y. Liang, L. Ma, and Y. He, "Msmote: Improving classification performance when training data is imbalanced," *Second International Workshop on Computer Science and Engineering*, October 2009.

[15] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD Explorations Newsletter - Special issue on learning from imbalanced datasets*, vol. 6, no. 1, pp. 20–29, June 2004.

[16] K. M. Ting, "An instance-weighting method to induce cost-sensitive trees," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 3, pp. 659–665, 2002.

[17] Z.-H. Zhou and X.-Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 1, pp. 63–77, 2006.

[18] J. R. Quinlan *et al.*, "Bagging, boosting, and C4. 5," in *AAAI/IAAI, Vol. 1*, 1996, pp. 725–730.

[19] R. Maclin and D. Opitz, "An empirical evaluation of bagging and boosting," *AAAI/IAAI*, vol. 1997, pp. 546–551, 1997.

[20] D. Farid, H. H. Nguyen, J. Darmont, N. Harbi, and M. Z. Rahman, "Scaling up Detection Rates and Reducing False Positives in Intrusion Detection using NBTree," in *International Conference on Data Mining and Knowledge Engineering (ICDMKE 2010)*, 2010, pp. 186–190.

[21] M. Galar, A. Fernández, E. Barrenechea, and F. Herrera, "EUSBoost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling," *Pattern Recognition*, vol. 46, no. 12, pp. 3460–3471, December 2013.

[22] S.-J. Yen and Y.-S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," *Expert Systems with Applications*, vol. 36, no. 3 (Part 1), pp. 5718–5727, April 2009.

[23] R. Barandela, R. Valdovinos, and J. S'anchez, "New applications of ensembles of classifiers," *Pattern Analysis and Applications*, vol. 64, no. 3 (Part 1), pp. 245–256, December 2003.

[24] S. Wang and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," *IEEE Symposium Series on Computational Intelligence and Data Mining(IEEE CIDM 2009)*, pp. 324–331, 2009.

[25] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," *European conference on computational learning theory*, vol. 1997, pp. 23–37, 1995.

[26] C. Seiffert, T. M. Khoshgoftaar, J. V. Hulse, and A. Napolitano, "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, January 2010.

[27] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: Improving Prediction of the Minority Class in Boosting," *7th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 107–109, 22-26 September 2003.

[28] V. Hulse, T. M. Khoshgoftaar, and A. Napolitano, "Experimental perspectives on learning from imbalanced data," *in Proc. 24th Int. Conf. Mach. Learn. , Corvallis, OR*, pp. 935–942, June 2007.

[29] A. A. Afza, D. M. Farid, and C. M. Rahman, "A Hybrid Classifier using Boosting, Clustering, and Naïve Bayesian Classifier," *World of Computer Science and Information Technology Journal (WCSIT) ISSN: 2221-0741 Vol*, vol. 1, pp. 105–109, 2011.

[30] D. M. Farid, A. Nowe, and B. Manderick, "A feature grouping method for ensemble clustering of high-dimensional genomic big data," in *Future Technologies Conference (FTC)*. IEEE, 2016, pp. 260–268.

Appendix A

Sample Source Code

A.1 Model source code

```
# Load Library
import numpy as np
from sklearn.preprocessing import label_binarize
import data_process as dp
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import roc_curve, auc, roc_auc_score
from sklearn.cluster import MiniBatchKMeans, KMeans
from sklearn.datasets.samples_generator import make_blobs
import random

fileName = 'pima' #iris pima wine habermans
loadDataset = dp.fileName(fileName)
classColumn = dp.classColumn(fileName)
idColumn = dp.idColumn(fileName)

AllData = np.array(loadDataset)
classType = AllData[:, classColumn]

if(idColumn != 'noId'):
    AllData = np.delete(AllData, idColumn, axis=1)

AllDataWithoutLabel = np.delete(AllData,-1,1)
X = AllDataWithoutLabel

# print(classType)
```

```
# print(X)

# Load data from sklearn incase of local data not available,
# Run either Local or Sklearn Dataset
# iris = dp.loadDataset()
# print(iris.data)

# Load CSV from URL using NumPy
# numpyData = dp.loadNumpy()
# print(numpyData)

# Class/label conversion
# Custom funcation that convert category to numerice number
y = dp.catLabelNumeric(classType)
# print(y)

# Majority and Minority data separation.
majorityClass = dp.find_majority(classType)
majorityClassName = dp.find_majority_class(classType)

majorityClassArray = AllData[classType == majorityClassName]
majorityWithoutLabel = np.delete(majorityClassArray,-1,1)

minorityClassArray = AllData[classType != majorityClassName]
minorityWithoutLabel = np.delete(minorityClassArray,-1,1)
print("Total Instances =", len(AllData),", Majority Instances =", len(
    majorityClassArray),
      "Minority Instances =", len(minorityClassArray))

#dp.dataPlot(X, y)

# data split into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
# print(X_train, len(X_train))

# silhouette_score calculation
noOfCluster = dp.silhouetteScore(X_train, y_train, minCluster = 3)
# print(noOfCluster)'kx'

#clustering
k_means = KMeans(init = "k-means++", n_clusters = noOfCluster)
k_means.fit(majorityWithoutLabel)
centroids = k_means.cluster_centers_
kmeansLabel = k_means.labels_
# print(kmeansLabel)
```

```
# print(centroids)

#cluster data plot
#dp.dataPlot(X_train, y_train, centroids)
#dp.dataPlot(X_train, kmeansLabel, centroids)
```

```
import pandas as pd
import numpy as np
import sklearn
from sklearn import preprocessing
from sklearn.cluster import MiniBatchKMeans, KMeans
from sklearn.metrics import roc_curve, auc, roc_auc_score,
    silhouette_samples, silhouette_score
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
from sklearn import datasets
import random

def fileName(name):
    return pd.read_csv("../dataset/"+name+".csv")

def classColumn(name):
    if(name == 'iris'):
        return 4 # total data = 151 , Majority =
    elif(name == 'pima'):
        return 8 #id # total data = 332 , Majority =
    elif (name == 'wine'):
        return 13 #id # total data = 179 , Majority =
    elif (name == 'haberman'):
        return 3
    elif (name == 'wisconsin'):
        return 10

def idColumn(name):
    if (name == 'pima'):
        return 0
    else:
        return 'noId'

def catLabelNumeric(y):
    le = preprocessing.LabelEncoder()
    le.fit(y)
    list(le.classes_)
    label = le.transform(y)
```

```
    #label = le.inverse_transform(y)

    return label

def ClusterIndicesNumpy(clustNum, labels_array): #numpy
    #https://stackoverflow.com/questions/36195457/python-sklearn-kmeans-how-to-get-the-values-in-the-cluster
    return np.where(labels_array == clustNum)[0]

def ClusterIndicesComp(clustNum, labels_array): #list comprehension
    return np.array([i for i, x in enumerate(labels_array) if x == clustNum])

def silhouetteScore(X_train, y_train, minCluster):
    # silhouette_score calculation
    noOfCluster = 0
    for n_clusters in range(minCluster, 20):
        k_means = KMeans(n_clusters = n_clusters)
        k_means.fit_predict(X_train)
        kmeansLabel = k_means.labels_
        silhouette_avg = silhouette_score(X_train, y_train)
        if(silhouette_avg >= noOfCluster):
            noOfCluster = n_clusters
    return noOfCluster;

def loadDataset():
    iris = datasets.load_iris()
    #X, y = iris.data, iris.target
    return iris

def loadNumpy():
    # Load CSV from URL using NumPy
    from urllib.request import urlopen
    url = "http://archive.ics.uci.edu/ml/machine-learning-databases/cmc/cmc.data"
    raw_data = urlopen(url)
    numpyData = np.loadtxt(raw_data, delimiter=",")
    # X, y = AllData[:,0:9], AllData[:,9]
    return numpyData

def informativeData(k_means, clusterItems, clusterNumber, quantity):
    result = []
    d = k_means.transform(clusterItems)[:, clusterNumber]
    closest = np.argsort(d)[::][:quantity]
    longest = np.argsort(d)[::-1][:5]
```

```
81      result = list(closest) + list(longest)
82      cd = clusterItems[closest]
83      return cd
84
85  def randomForest(MeargeMajMin, y_train):
86      clf = RandomForestClassifier(max_depth=2, random_state=0, class_weight
        ="balanced")
87      clf.fit(MeargeMajMin, y_train)
88      return clf
89
90  def rocAuc(y_test, y_pred):
91      fpr, tpr, thres = roc_curve(y_test, y_pred, pos_label=0)
92      roc_auc = auc(fpr, tpr)
93
94      plt = rocPlot(fpr, tpr, roc_auc)
95
96      return roc_auc
97      #To return individual roc_auc score or a clusters
98      #return roc_auc
99
100 def rocPlot(fpr, tpr, roc_auc):
101     plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area =
        %0.2f)' % roc_auc)
102     plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
103     plt.xlim([0.0, 1.0])
104     plt.ylim([0.0, 1.05])
105     plt.xlabel('False Positive Rate')
106     plt.ylabel('True Positive Rate')
107     plt.title('Receiver operating characteristic example')
108     plt.legend(loc="lower right")
109     return plt.show()
110
111 def dataPlot(data, label, centroids = None):
112     for i in set(label):
113         index = label == i
114         plt.plot(data[index,0], data[index,1], 'o')
115         if centroids is not None:
116             lines = plt.plot(centroids[i,0], centroids[i,1], 'kx')
117             # make the centroid x's bigger
118             plt.setp(lines, 'color', 'r', ms=15.0)
119             plt.setp(lines, 'color', 'r', mew=2.0)
120
121     plt.show()
122
123 def find_majority(k):
```

```
124     myMap = {}
125     maximum = ( '', 0 ) # (occurring element, occurrences)
126     for n in k:
127         if n in myMap: myMap[n] += 1
128         else: myMap[n] = 1
129 
130         # Keep track of maximum on the go
131         if myMap[n] > maximum[1]: maximum = (n,myMap[n])
132 
133     return maximum
134 
135 def find_majority_class(k):
136     myMap = {}
137     maximum = ( '', 0 ) # (occurring element, occurrences)
138     for n in k:
139         if n in myMap: myMap[n] += 1
140         else: myMap[n] = 1
141 
142         # Keep track of maximum on the go
143         if (myMap[n] > maximum[1]):
144             maximum = (n,myMap[n])
145             majorityClassName = n
146 
147     return majorityClassName
148 
149 
150 def cluster_center(myData, numberOfCluster):
151     labels = [x[0] for x in myData]
152     # a = np.array([x[1:] for x in myData])
153     a = np.array([x[0:] for x in myData])
154     clust_centers = numberOfCluster
155 
156     model = sklearn.cluster.k_means(a, clust_centers)
157     # print(a)
158     return dict(zip(labels, model[1])) #INSTANCE AND LABEL
159     # return model[1] #RETURN LABEL
```