Heaven's Light is Our Guide

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Rajshahi University of Engineering & Technology, Bangladesh

Clustering-based Undersampling Using K-Means in Class-Imbalanced Dataset

Authors
Rifat Bin Siraj
Roll No. 1503024
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology

Supervised by
Emrana Kabir Hashi
Assistant Professor
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology

ACKNOWLEDGEMENT

At first, we would like to thank the Almighty Allah for giving us the opportunity and enthusiasm along the way for the completion of our thesis work.

We would like to express our sincere appreciation, gratitude, and respect to our supervisor Emrana Kabir Hashi, Assistant Professor of Department of Computer Science and Engineering, Rajshahi University of Engineering and Technology, Rajshahi. Throughout the year she has not only given us technical guidelines, advice, and necessary documents to complete the work she has also given us continuous encouragement, advice, helps, and sympathetic co-operation whenever she deemed necessary. Her continuous support was the most successful tool that helped us to achieve our results. Whenever we were stuck in any complex problems or situation she was there for us at any time of the day. Without her sincere care, this work not has been materialized in the final form that it is now at the present.

I am also grateful to all the respective teachers of Computer Science and Engineering, Rajshahi University of Engineering and Technology, Rajshahi for good & valuable suggestions and inspirations from time to time.

Finally, I convey my thanks to my parents, friends, and well-wishers for their constant inspirations and many helpful aids throughout this work.

Date: 13/12/2020 Rifat Bin Siraj

RUET, Rajshahi

Heaven's Light is Our Guide

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Rajshahi University of Engineering & Technology, Bangladesh

CERTIFICATE

This is to certify that this thesis report entitled ***"Clustering-based undersampling in class-imbalanced data"*** *submitted by* ***Rifat Bin Siraj, Roll:1503024*** *in partial fulfillment of the requirement for the award of the degree of Bachelor of Science in Computer Science & Engineering of Rajshahi University of Engineering & Technology, Bangladesh is a record of the candidate own work carried out by him under my supervision. This thesis has not been submitted for the award of any other degree*

Supervisor

Emrana Kabir Hashi
Assistant Professor
Department of Computer Science &Engineering
Rajshahi University of Engineering &Technology
Rajshahi-6204

External Examiner

ABSTRACT

Class Imbalance problem is common in various real-world datasets. In a binary classification problem where the whole dataset divides into two classes. One of them is called a majority class and another is called a minority class. In an imbalanced dataset problem, the majority class contains a greater number of data points than the minority class. So if someone applies a classifier in this type of dataset it will predict almost everything in the majority class. In this experiment, the class imbalance problem has been tried to solve in a data level solution. The approach in this experiment is at first divide the dataset into two classes then solve the imbalance problem by using K-means in the majority class where the number of clusters is equal to the number of data points in the minority class. Then vary the cluster(-5,-10,+5,+10) and calculate the performance evolution matrix. The classifier in this experiment will use SVM (Support Vector Machine). When K=N then the Accuracy, Recall, Specificity, F1-score are maximum, and when K=N+5 misclassification rate is maximum and when K=N-5, Precision is maximum. And when K=N then the misclassification rate is minimum which is our goal. These experimental results are obtained using 1 large scale dataset(Protein homology prediction). The imbalance ratio of the dataset is 111.46.

CONTENTS

Page No.

ACKNOWLEDGEMENT i

CERTIFICATE ii

ABSTRACT iii

CHAPTER 1

Introduction 1-10

1.1 Problem Statement……1

1.2 Class imbalance Problem……..3

1.2.1Solutions for data sets with class imbalance…….4

1.3 Literature Review……..5

1.3.1 Based Paper……..5

1.3.2 Related Work ……...7

1.4 Motivation ……8

1.5 Thesis Contribution ……...9

1.6 Thesis Organization ……...9

1.7 Conclusion ……10

CHAPTER 2

Background Studies 11-22

2.1 Clustering Based Under-sampling……11

2.2 Clustering Based Over-sampling …….12

2.3 K-Means algorithm ……..13

2.4 Support Vector Machine ……..17

CHAPTER 3

Methodology **23-26**

3.1 Existing Model ……23

3.2 Proposed Model ……...25

CHAPTER 4

Experimental Results and Performance Analysis **27-41**

4.1 Data Set …… 27

4.2 Procedure……29

4.3 Performance Evolution Matrix……30

4.4 Experimental Result……32

4.5 Comparison ……39

4.5.1 Result Comparison…….39

4.6 Conclusion ……..41

CHAPTER 5

Conclusion and Future Works **42-44**

9.1 Summary ……..42

9.2 Applications …….43

9.3 Limitations ……...44

9.4 Future Work …….44

9.5 Conclusion……44

REFERENCES **45-46**

LIST OF TABLES

Table Number	Table Title	Page No.
4.1	dataset information ……28	
4.2	Performance evolution of Proposed Model ……...33	
4.3	Competitive result between two models ……... 39	

LIST OF FIGURES

Figure Number	Figure Caption	Page No.
1.1	Balance & Imbalance dataset…….2	
1.2	Example of class overlapping……3	
1.3	Example of Small disjuncts ……..4	
1.4	Confusion matrix for class Miss-Classification ……...6	
2.1	Clustering Based Under-sampling Method……...12	
2.2	Clustering Based Over-sampling Method ……..13	
2.3	Visualization of K-Means algorithm ……..16	
2.4	Linear SVM…….18	
2.5	Non-linear SVM using RBF …….19	

Figure Number	Figure Caption	Page No.
3.1	Existing Model for Clustering-based Under-sampling ……..….24	
3.2	Proposed Model for Clustering-based Under-sampling ……...25	
4.1	Protein Homologous Prediction dataset……29	
4.2	Confusion Matrix …….30	
4.3	Accuracy of Proposed model……33	
4.4	Misclassification Rate of proposed Model ……..34	
4.5	Precision of proposed model ……35	
4.6	Recall of proposed model …….36	
4.7	Specificity of proposed model ……..37	
4.8	F1-score of proposed model …….38	
4.9	Accuracy difference between previous and proposed model …….40	

CHAPTER 1

Introduction

This chapter begins with the motivation behind this thesis topic. Then the literature reviews are discussed right after. Then the discussion of what is a class imbalance problem and different type of solution to it. Then, in thesis contribution, contributions of the thesis are outlined. Finally, the chapter ends with a conclusion.

1.1 Problem Statement

Real-world datasets mainly show the particularity to have several samples of a given class. The class under-represented compared to the other class. In machine learning and data mining, it is very to train an effective learning model if the class distribution in a given training dataset is an imbalance. This imbalance gives rise to the "class imbalance" problem. One class might have a large number of data examples, whereas the other might have only a few. For most of the data mining algorithms, the rare objects are much more difficult to identify than the common objects.

The class imbalance problem has been encountered in multiple areas such as telecommunication management, bio-informatics, fraud detection, and medical diagnosis, and has been considered one of the top 10 problems in data mining and pattern recognition. Imbalanced data substantially compromises the learning process, since most of the standard machine learning algorithms expect balanced class distribution or an equal miss-classification cost. Therefore the primary class of interest in data mining is usually the minority class.

Without consideration of the class imbalance problem, learning algorithms or constructed models can be submerged by the majority class and can ignore the minority class. For example,

Consider a two-class data set with an imbalance ratio of 99%, where the majority class containing 99% of the data set and the minority class contains only 1%. To minimize the error rate, the learning algorithm classifies all of the examples into the majority class, which yields an error rate of 1%. In this case, all of the examples belonging to the minority class are paramount and must be identified as incorrectly classified Bankruptcy prediction is a practical class imbalance problem. In particular, the numbers of bankruptcy cases (i.e. the minority class) are usually much smaller than those of non-bankruptcy cases (i.e. the majority class). The type I error rate, which means that a prediction model incorrectly classifies the bankruptcy case into the non-bankruptcy class, is more critical than the average rate of classification accuracy. This is because higher type I error rates are likely to increase bad debts for financial institutions. A variety of methods have been proposed to solve this problem. Such methods can be divided into four types: algorithmic-level methods, data-level methods, cost-sensitive methods, and ensembles of classifiers. In particular, the data-level methods, which focus on preprocessing the imbalanced data sets before constructing the classifiers, are widely considered in the literature. This is because the data preprocessing and classifier training tasks can be performed independently.[4]

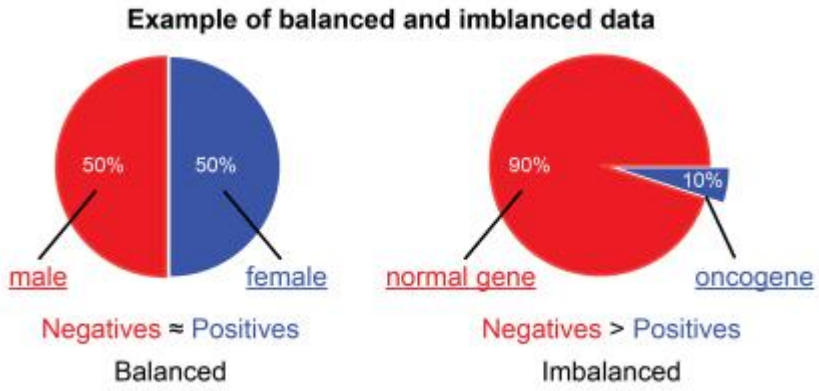

Figure 1.1:Balance & Imbalance dataset[a]

[a]-https://medium.com/analytics-vidhya/what-is-balance-and-imbalance-dataset-89e8d7f46bc5[03-12-2020]

In figure 1.1 showing the difference between class imbalanced and class balanced dataset. In a balanced dataset, the positive value and negative values are equal and in the imbalanced dataset, the number of positives and negatives are not equal.

For this reason, several approaches have been specifically proposed to handle such datasets.

1.2 Class Imbalance problem

Class imbalance is a problem in data sets with skewed distributions of data points. This has the following characteristics

• **Class overlapping**: When the data samples belonging to different classes overlap (as shown in Fig.1.2), classifiers have difficulty effectively distinguishing between different classes. In most cases, instances belonging to the minority class are classified into the majority class.[4]

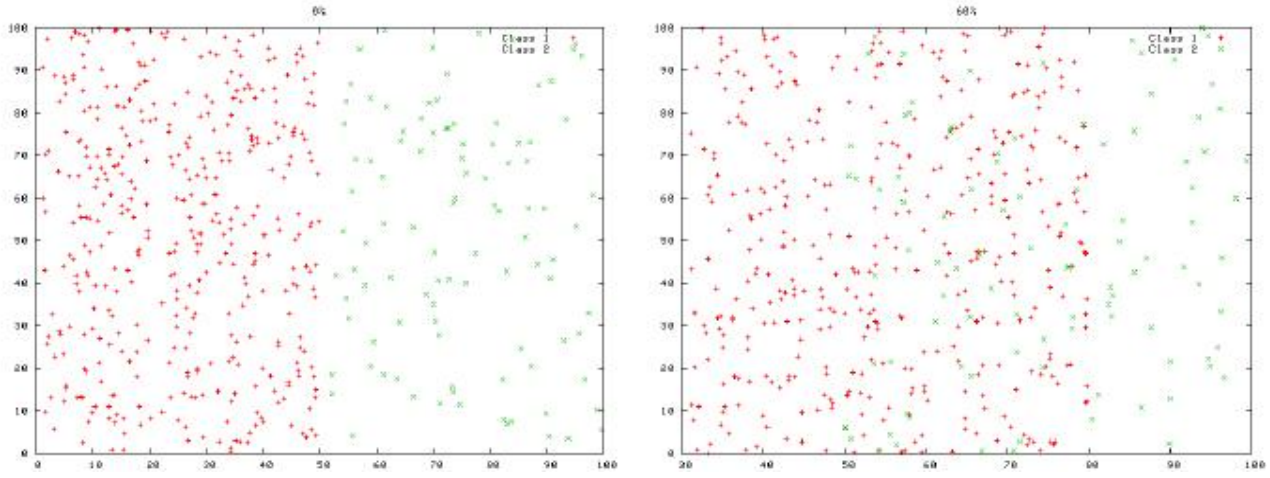

Figure 1.2:Example of class overlapping[2]

In figure 1.2 showing the example of class overlapping in the class imbalance problem. Here the first picture shown no class overlapping because the data point red and green are

separated but in the second picture, there are some green data points in the red region and some red datapoint in the green region.

• **Small sample size**: In practice, collecting sufficient data for class imbalanced data sets is challenging. One solution is to balance the imbalance ratios of the data sets to reduce the miss-classification error. [4]

• **Small disjuncts**: The data samples in the minority classes are distributed in numerous feature spaces, as shown in Fig.1.3. This causes a high degree of complication during the classification stage. Due to a significant difference between the sample sizes of two different classes (i.e. high imbalance ratios), classifiers may treat some of the data points in the minority class as outliers, which produces a very high miss-classification error rate For the minority class. As data set sizes become increasingly larger in numerous real-world applications such as medical decision-making, fault diagnosis, and face recognition, the consequences of class imbalance problems become greater. [4]

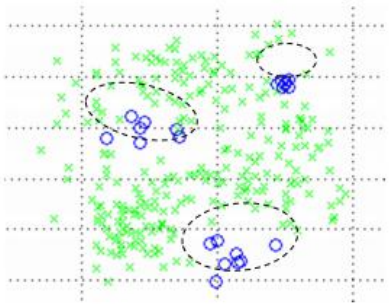

Figure 1.3:Example of Small disjuncts [3]

in figure 1.3 green data points are not evenly distributed. So it occurs problem for the classifier.

1.2.1 Solutions for data sets with class imbalance

In general, three types of approaches can solve the class imbalance problem. These solutions are based on data, algorithms, or cost sensitivity. Recently, ensembles of classifiers have been employed to handle imbalanced data sets with data-based approaches.

A.Data-level solutions

The Data-level solutions are based on preprocessing (or balancing) the collected imbalanced training data set by either under-sampling or oversampling strategies. The under-sampling approaches are used to reduce the data samples in the majority class, whereas the oversampling approaches are used to increase the data samples in the minority class. [4]

The advantage of these approaches is to make the sampling and classifier training processes independent.

B. Algorithm-level solutions

The algorithm-level solutions involve proposing novel algorithms or modifying existing algorithms to directly handle data sets with class imbalance; such algorithms can outperform previously existing algorithms. The threshold method and one-class learning method are widely used approaches in the literature. The threshold method involves setting different threshold values for different classes during the classifier learning stage, whereas the one-class learning method entails training the classifier from a training set that contains only one specific class. Other types of algorithms, such as evolving clustering in neuro-fuzzy systems, evolving clustering of dynamic data in spiking neural networks, clustering personalized modeling, and clustering through quantum-inspired evolutionary algorithms, have also been developed to deal with imbalanced data.[4]

C.Cost-sensitive solutions

The cost-sensitive solutions focus on defining different miss-classification costs of classifiers for different classes. Then, a confusion matrix for the miss-classification cost can be produced, as shown in Fig 1.4.

		Prediction	
		Class i	Class j
True	Class i	0	λ_{ij}
	Class j	λ_{ji}	0

Figure 1.4:Confusion Matrix for class Miss-classification[4]

The cost for correct classification is 0. If the data sample whose true class is *j* is incorrectly classified into the *i* class, its miss-classification cost is *λij*. Therefore, according to Equation.

Figure 1.4 shown the confusion matrix for miss-classification.

The miss-classification cost is

$$R(a_i \mid x) = \sum_i \lambda_{ij} P(v_j \mid x) \tag{1.1}$$

1.3 Literature Review

1.3.1 Based Paper

Clustering-based under-sampling in class-imbalanced data which is written by Wei-Chao Lin, Chih-Fong Tsai, Ya-Han Hu, Jing-Shang Jhang[4] discusses the imbalance problem briefly. The solution they gave in this paper is the data level solution. The work procedure of the existing model is given below

- First divided the dataset into train and test sets.
- Defined the train set into majority and minority classes.
- Using under-sampling methods in the majority class and then balance the dataset with minority class.
- Train with a different type of er classifier with the newly formed balance dataset.
- Test the dataset and compare the accuracy.

The main limitation of this paper is that all the analysis is found by Weka software. So the result maybe not as accurate as it should be if we use row code. The goal of this experiment is to compare the result between row code and software-generated accuracy.

1.3.2 Related Work

On the class imbalance problem which is written by Xinjian Guo, Yilong Yin, Cailing Dong, Gongping Yang, Guangtong Zhou [8] discuss the overall class imbalance problem and there overall all the possible solutions.

Under-sampling class imbalanced datasets by combining clustering analysis and instance selection which is written by Chih-Fong Tsai , Wei-Chao Lin, Ya-Han Hu, Guan-Ting Yao [9] describes the data level solution ensemble methods. The ensemble method is the combination of different types of classifiers together.

A novel approach for solving skewed classification problem using cluster based ensemble method which is written by Rekha, Gillala, V. Krishna Reddy, and Amit Kumar Tyagi[7] discuss the data level solution of the class imbalance problem. In their work, they used ensemble methods like AdaBoost, RUSBoost, and SMOTEBoost.

Undersampled K-means approach for handling imbalanced distributed data which is written by Kumar, N. Santhosh, et al.[10] describes the solution to the imbalance dataset problem. And the solution they have to evaluate as a data level solution is the K-means algorithm.

Cluster-based under-sampling approaches for imbalanced data distributions which are written by Yen, Show-Jane, and Yue-Shi Lee [5] discuss the data level solution of the class imbalance problem. Their approach first clusters all the training samples into some clusters. The main idea is that there are different clusters in a dataset, and each cluster

seems to have distinct characteristics. If a cluster has more majority class samples and fewer minority class samples, it will behave like the majority class samples. On the other hand, if a cluster has more minority class samples and fewer majority class samples, it doesn't hold the characteristics of the majority class samples and behaves more like the minority class samples. So they select a suitable number of majority class samples from each cluster by considering the ratio of the number of majority class samples to the number of minority class samples in the cluster.

Learning from imbalanced data using ensemble methods and cluster-based undersampling which is written by Sobhani, Parinaz, Herna Viktor, and Stan Matwin[12] & Clustering-based under-sampling for improving speaker verification decisions using AdaBoost which is written by Altınçay, Hakan, and Cem Ergün[15] explain the data level ensemble method for handling the class imbalance problem.

1.4 Motivation

Class imbalanced datasets occur in many real-world applications where the class distributions of data are highly imbalanced. For the two-class case, without loss of generality, one assumes that the minority or rare class is a positive class, and the majority class is a negative class. Often the minority class is very infrequent, such as 1 % of the dataset. If one applies most traditional (cost-insensitive) classifiers on the dataset, they are likely to predict everything as negative (the majority class). This was often regarded as a problem in learning from highly imbalanced datasets. The motivation behind the problem if someone uses any classifier algorithm in any class imbalanced dataset then the classifier will predict almost everything in the majority class because the number of a datapoint is much bigger than the datapoint in the minority class. So It should be solved for better prediction in machine learning. Many real-world problems occur because of the class imbalance dataset. It is very difficult to

collect a balanced dataset. Data level solutions are much easier than other solutions. So in this experiment focused on data level solution.

1.5 Thesis Contribution

The main contributions of the thesis are as follows:

- This work would try to implement a model to solve the data level class imbalance problem.
- For this model, we will follow clustering-based under-sampling methods.
- This work would try to reduced the majority class datapoint. Then try to balance with the minority class then.
- For clustering-based under-sampling, we will use the K-means clustering algorithm
- For the train and test set, we will use the Support Vector Machine(SVM) and try to calculate the accuracy.
- This work would like to compare the result with the previous one.

1.6 Thesis Organization

The rest of the thesis is organized as follows:

Chapter 2- Background Studies.

This chapter describes the basic idea of clustering based undersampling. The unsupervised machine learning technique K-means clustering algorithm and Support Vector Machine (SVM) and how Those methods can be used as a clustering-based under-sampling method.

Chapter 3 – Methodology

This chapter describes the existing under-sampling methods and the proposed under-sampling methods.

Chapter 4 – Experimental Result Analysis

This chapter discusses the dataset we worked with and the findings, results, and analysis of the thesis work. It also compares with other works relating to this area.

Chapter 5 – Conclusion and Future Works

This chapter concludes the thesis, describes its limitation, and shows a direction for future work.

1.7 Conclusion

The class imbalance problem is a common problem in machine learning. This is a problem encountered in numerous real-world applications such as medical diagnosis, financial crisis prediction, and e-mail filtering.

CHAPTER 2

Background Studies

This chapter covers all the necessary algorithm and theories to handle the class imbalance problem. The discussion begins with clustering based under- sampling then clustering-based oversampling. Then next the discussion about the K-means algorithm SVM algorithm was also placed. At last, the chapter ends with the conclusion.

2.1 Clustering Based Under-sampling

Figure 2.1 shows the procedure for clustering-based under-sampling. The processes are described as follows. Given an imbalanced data set *D* composed of a majority class and a minority class, the majority and minority classes contain M and N data points, respectively.

- The first step is to divide this imbalanced data set into training and testing sets based on the *k*-fold cross-validation method
- .
- The second step is to divide the training set into a majority class subset and a minority class subset.
- The third step is the clustering-based under-sampling method is employed to reduce the number of data samples in the majority class.
- The fourth step to reduced the majority class subset is then combined with the minority class subset, resulting in a balanced training set.
- Finally, the classifier is trained and tested by the balanced training and testing

In contrast to the well-known methods described in, random under-sampling is performed without considering any machine-learning-based under-sampling method.[4]

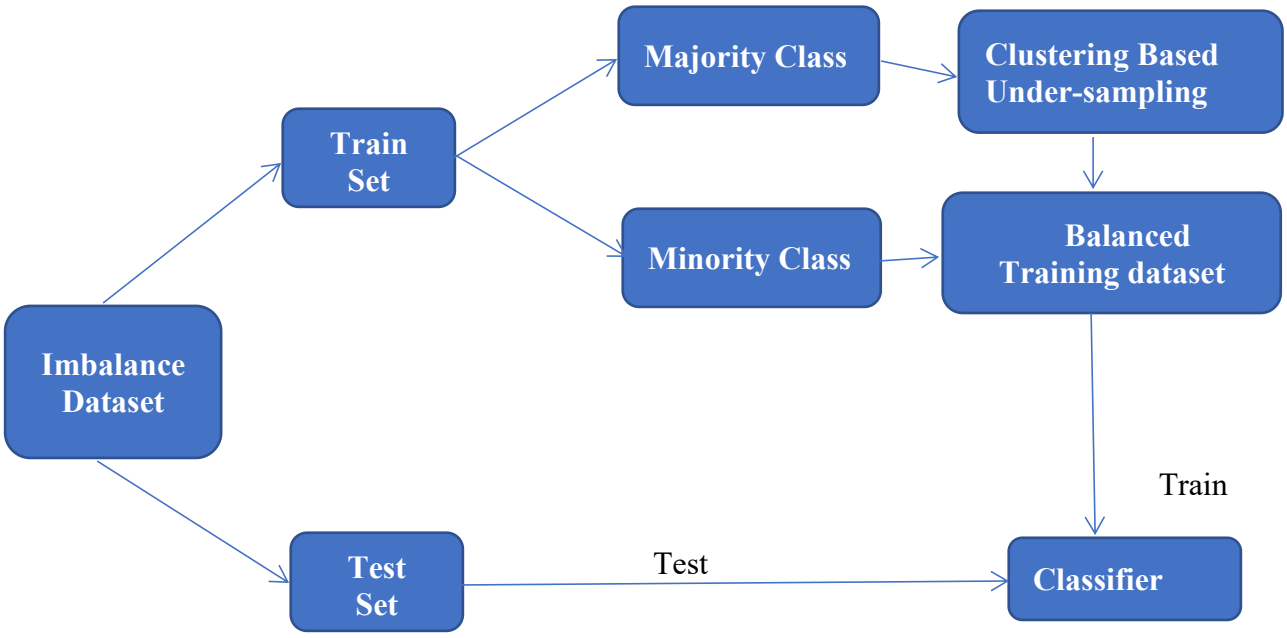

Figure 2.1: Clustering Based Under-sampling Method

2.2 Clustering Based Over-sampling

Figure 2.2 shows the procedure for clustering-based over-sampling. The processes are described as follows. Given an imbalanced data set *D* composed of a majority class and a minority class, the majority and minority classes contain M and N data points, respectively.

- The first step is to divide this imbalanced data set into training and testing sets based on the *k*-fold cross-validation method
- .
- The second step is to divide the training set into a majority class subset and a minority class subset.
- The third step is the clustering-based Over-sampling method is employed to increase the number of data samples in the minority class.
- The fourth step to increased minority class subset is then combined with the majority class subset, resulting in a balanced training set.
- Finally, the classifier is trained and tested by the balanced training and testing

In contrast to the well-known methods described in, random over-sampling is performed without considering any machine-learning-based over-sampling method.

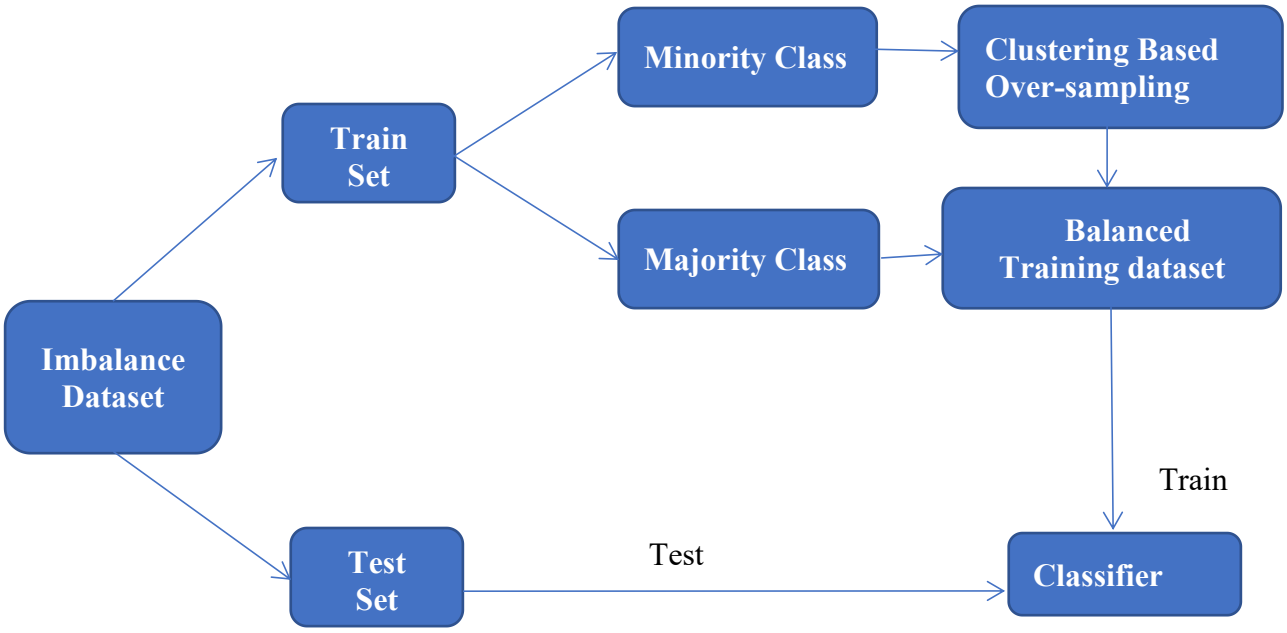

Figure 2.2: Clustering Based Over-sampling

2.3 K-Means Algorithm [a]

Kmeans algorithm is an iterative algorithm that tries to partition the dataset into *K*pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to **o**nly one group. It tries to make the intra-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) data points are within the same cluster.

[a]-https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a

The way the K-means algorithm works is as follows:

i. Specify the number of clusters *K*.

ii. Initialize centroids by first shuffling the dataset and then randomly selecting *K* data points for the centroids without replacement.

iii. Keep iterating until there is no change to the centroids. i.e the assignment of data points to clusters isn't changing.

- Compute the sum of the squared distance between data points and all centroids.
- Assign each data point to the closest cluster (centroid).
- Compute the centroids for the clusters by taking the average of all data points that belong to each cluster.

The approach k-means follows to solve the problem is called Expectation-Maximization. The E-step is assigning the data points to the closest cluster. The M-step is computing the centroid of each cluster. Below is a break down of how we can solve it mathematically.

The objective function is:

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} w_{ik} \, || \, x^i - \mu_k \, ||^2 \tag{2.1}$$

where $w_{ik} = 1$ for data point xi if it belongs to cluster *k*; otherwise $w_{ik} = 0$. Also, μ_k is the centroid of xi's cluster.

It's a minimization problem of two parts. We first minimize J w.r.t. wik and treat μ_k fixed. Then we minimize J w.r.t. μ_k and treat w_{ik} fixed. Technically speaking, we differentiate J w.r.t. w_{ik} first and update cluster assignments. Then we differentiate J w.r.t. μ_k and re$_c$ompute the centroids after the cluster assignments from the previous step.

Therefore, E-step is:

$$\frac{\partial j}{\partial w_{ik}} = \sum_{i=1}^{m}\sum_{k=1}^{K} \| x^i - \mu_k \|^2 \tag{2.2}$$

$$\Rightarrow w_{ik} = \begin{cases} 1 & \text{If k=} \arg\min_j \| x^i - \mu_j \|^2 \\ 0 & \text{Otherwise} \end{cases} \tag{2.3}$$

In other words, assign the data point x_1 to the closest cluster judged by its sum of squared distance from the cluster's centroid.

And M-step is:

$$\frac{\partial J}{\partial \mu_k} = 2\sum_{i=1}^{m} w_{ik}(x^i - \mu_k) = 0 \tag{2.4}$$

$$\Rightarrow \mu_k = \frac{\sum_{i=1}^{m} w_{ik} x^i}{\sum_{i=1}^{m} w_{ik}} \tag{2.5}$$

This translates to recomputing the centroid of each cluster to reflect the new assignments.

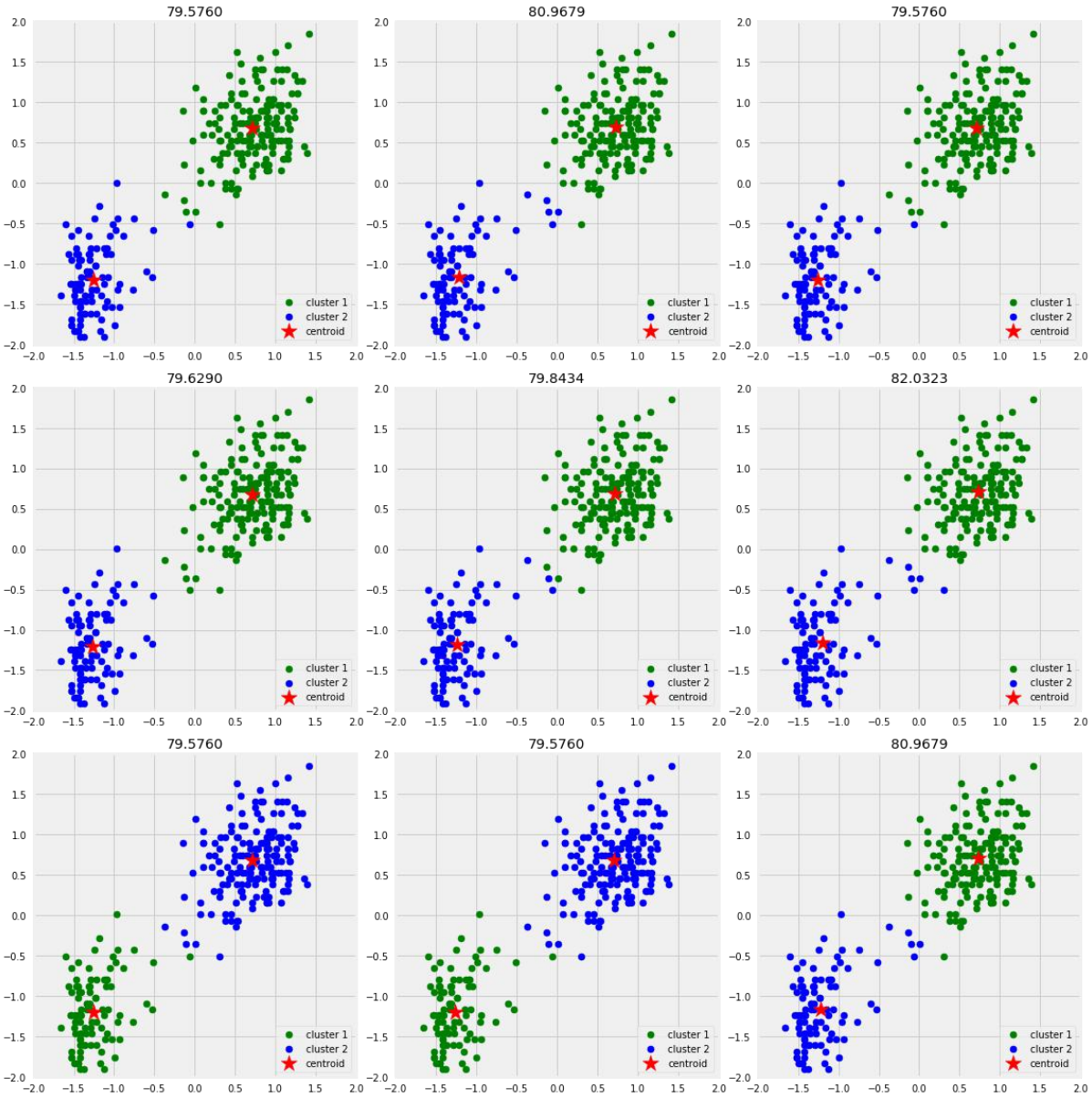

Figure 2.3:Visualization of K-Means algorithm[a]

In figure 2.3 showing the green and blue data point and red star as a cluster center. Kmeans have to find the exact center point in the last picture and the iterations are also shown in those pictures.

[a]-https://miro.medium.com/max/2400/1*smb3nXFMihSmbJGO3kS0Ww.png

2.4 Support Vector Machine(SVM)[a]

2.4.1 Theory

Support vector machines are a set of supervised learning methods used for classification, regression, and outliers detection. All of these are common tasks in machine learning.

We can use them to detect cancerous cells based on millions of images or you can use them to predict future driving routes with a well-fitted regression model.

There are specific types of SVMs you can use for particular machine learning problems, like support vector regression (SVR) which is an extension of support vector classification (SVC).

The main thing to keep in mind here is that these are just math equations tuned to give you the most accurate answer possible as quickly as possible.

SVMs are different from other classification algorithms because of the way they choose the decision boundary that maximizes the distance from the nearest data points of all the classes. The decision boundary created by SVMs is called the maximum margin classifier or the maximum margin hyperplane.

2.4.2 How an SVM works

A simple linear SVM classifier works by making a straight line between two classes. That means all of the data points on one side of the line will represent a category and the data points on the other side of the line will be put into a different category. This means there can be an infinite number of lines to choose from.

[a]-https://www.freecodecamp.org/news/svm-machine-learning-tutorial-what-is-the-support-vector-machine-algorithm-explained-with-code-examples/

What makes the linear SVM algorithm better than some of the other algorithms, like k-nearest neighbors, is that it chooses the best line to classify your data points. It chooses the line that separates the data and is the furthest away from the closet data points as possible.

A 2-D example helps to make sense of all the machine learning jargon. Basically, you have some data points on a grid. You're trying to separate these data points by the category they should fit in, but you don't want to have any data in the wrong category. That means you're trying to find the line between the two closest points that keeps the other data points separated.

So the two closest data points give you the support vectors you'll use to find that line. That line is called the decision boundary. Figure 2.4 have shown the linear SVM method. Here yellow and purple datapoint is separated by the line.

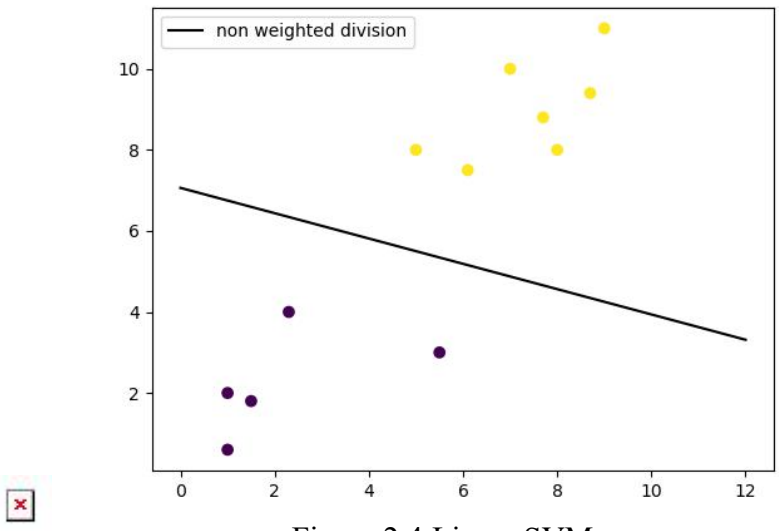

Figure 2.4:Linear SVM[a]

The decision boundary doesn't have to be a line. It's also referred to as a hyperplane because you can find the decision boundary with any number of features, not just two.

[a]-https://www.freecodecamp.org/news/svm-machine-learning-tutorial-what-is-the-support-vector-machine-

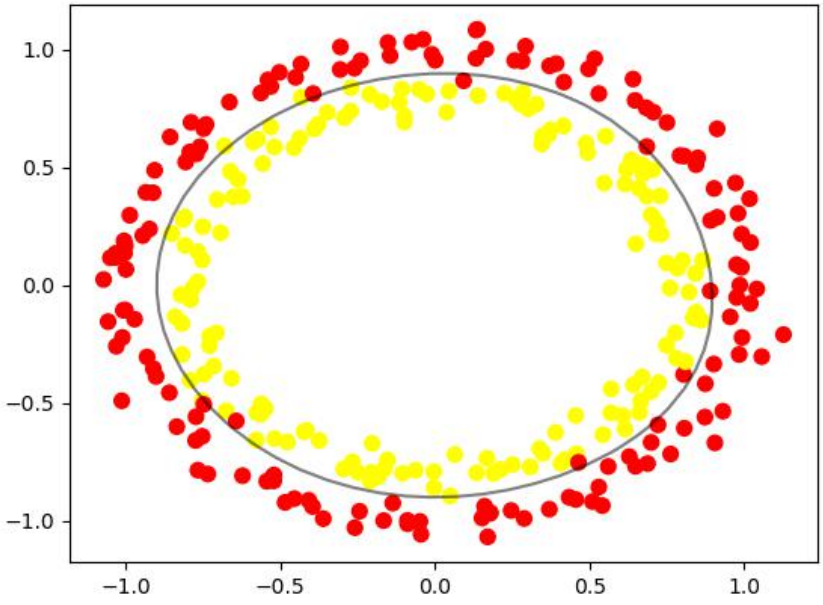

Figure 2.5:Non-linear SVM using RBF

In figure 2.5 have shown the non-linear SVM. Here yellow and red data points are non-linear datapoint.

2.4.3 Types of SVMs

There are two different types of SVMs, each used for different things:

i. Simple SVM: Typically used for linear regression and classification problems.

ii. Kernel SVM: Has more flexibility for non-linear data because you can add more features to fit a hyper-plane instead of a two-dimensional space.

2.4.4 Why SVMs are used in machine learning

SVMs are used in applications like handwriting recognition, intrusion detection, face detection, email classification, gene classification, and in web pages. This is one of the reasons we use SVMs in machine learning. It can handle both classification and regression on linear and non-linear data.

Another reason we use SVMs is that they can find complex relationships between your data without you needing to do a lot of transformations on your own. It's a great option

when you are working with smaller datasets that have tens to hundreds of thousands of features. They typically find more accurate results when compared to other algorithms because of their ability to handle small, complex datasets.

Here are some of the pros and cons of using SVMs.

Pros

- Effective on datasets with multiple features, like financial or medical data.
- Effective in cases where the number of features is greater than the number of data points.
- Uses a subset of training points in the decision function called support vectors which makes it memory efficient.
- Different kernel functions can be specified for the decision function. You can use common kernels, but it's also possible to specify custom kernels.

Cons

- If the number of features is a lot bigger than the number of data points, avoiding over-fitting when choosing kernel functions and regularization term is crucial.
- SVMs don't directly provide probability estimates. Those are calculated using an expensive five-fold cross-validation.
- Works best on small sample sets because of its high training time.
- Since SVMs can use any number of kernels, we must know about a few of them.

2.4.5 Kernel functions

A. Linear

These are commonly recommended for text classification because most of these types of classification problems are linearly separable.

The linear kernel works really well when there are a lot of features, and text classification problems have a lot of features. Linear kernel functions are faster than most of the others and you have fewer parameters to optimize.

Here's the function that defines the linear kernel:

$$f(X) = w^T * X + b \quad (2.6)$$

In this equation, **w** is the weight vector that you want to minimize, **X** is the data that you're trying to classify, and **b** is the linear coefficient estimated from the training data. This equation defines the decision boundary that the SVM returns.

B.Polynomial

The polynomial kernel isn't used in practice very often because it isn't as computationally efficient as other kernels and its predictions aren't as accurate.

Here's the function for a polynomial kernel:

$$f(X1, X2) = (a + X1^T * X2) ^ b \quad (2.7)$$

This is one of the more simple polynomial kernel equations you can use. **f(X1, X2)** represents the polynomial decision boundary that will separate your data. **X1** and **X2** represent your data.

C.Gaussian Radial Basis Function (RBF)

One of the most powerful and commonly used kernels in SVMs. Usually the choice for non-linear data.

Here's the equation for an RBF kernel:

$$f(X1, X2) = \exp(-gamma * ||X1 - X2||^2) \quad (2.8)$$

In this equation, **gamma** specifies how much a single training point has on the other data points around it. ||**X1 - X2**|| is the dot product between your features.

D.Sigmoid

More useful in neural networks than in support vector machines, but there are occasional specific use cases.

Here's the function for a sigmoid kernel:

$$f(X, y) = \tanh(\text{alpha} * X^T * y + C) \quad (2.9)$$

In this function, **alpha** is a weight vector and **C** is an offset value to account for some miss-classification of data that can happen.

E.Others

There are plenty of other kernels you can use for your project. This might be a decision to make when you need to meet certain error constraints, you want to try and speed up the training time, or you want to super tune parameters.

2.5 Conclusion

This chapter discussed all the necessary algorithm used in the experiment. This chapter will give the basic ideas of all the algorithms that need to be understood for this experiment. K-means is a very popular unsupervised classifier and SVM is a supervised classifier.

Chapter 3

Methodology

This chapter covers the previous model description and proposed model description. At last, the chapter ends with the conclusion. The previous model covers clustering-based under-sampling and its procedure. In proposed model section covers the changes that have been made to find the best accuracy.

3.1 Existing Model

Two strategies employing a clustering algorithm to under-sample the majority class data set are discussed. Note that although numerous clustering algorithms are mentioned in the literature, we consider only the *k*-means algorithm in this experiment because it is widely used and can thus be regarded as a baseline clustering method. The two strategies are described as follows.

- In the first strategy, the number of clusters (i.e. k) is set to be equal to the number of data samples in the minority class (i.e. $k = N$). Then, the k cluster centers (or centroids) are produced by the *k*-means algorithm over the M data samples in the majority class. These cluster centers are used to replace the entire majority class data set. Consequently, both the majority and minority class data sets contain the same number of data samples.

- In the second strategy, because each cluster center is the mean of the data samples in a cluster, it is a new additional data sample for the majority class. The nearest neighbor of each cluster center, which is a real data sample of M, is selected to replace the k cluster centers used in the first strategy. In particular, the Euclidean distance is used to measure the level of similarity between the cluster center and the

data samples in the same cluster. Therefore, the reduced majority class data set contains the same number of data samples as the minority class.

Although both strategies produce the same number of data samples to replace the *M* data samples in the majority class, the data points of both strategies in the feature space are somewhat different. A sensitivity study was conducted by using different numbers of *k* (i.e. $k \pm 5$ and $k \pm 10$, where *k* is *N*) with these two strategies. The results can clarify the differences in classification performance obtained by using different numbers of data samples in the majority class data set.

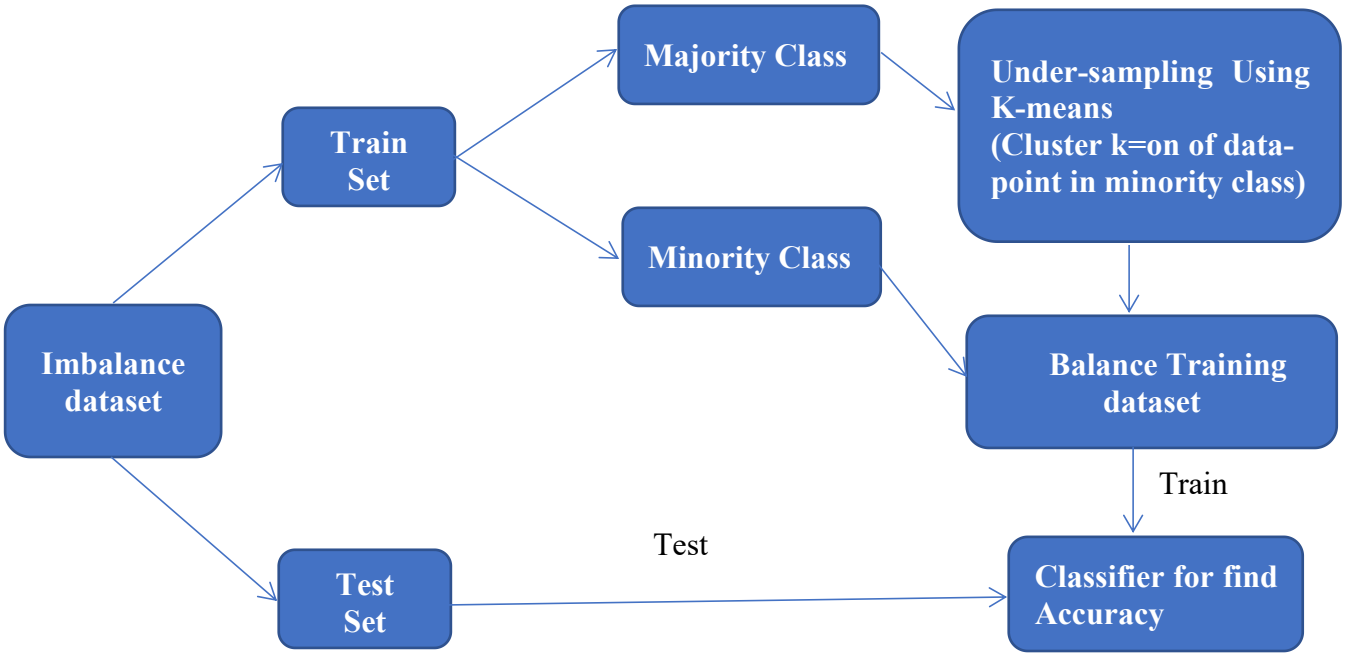

Figure 3.1: Existing Model for Clustering-based Under-sampling

3.2 Proposed Model

Suppose an imbalanced data set *D* is composed of a majority class and a minority class, the majority, and minority classes contain M and N data points, respectively. Then the steps are shown in figure 3.2

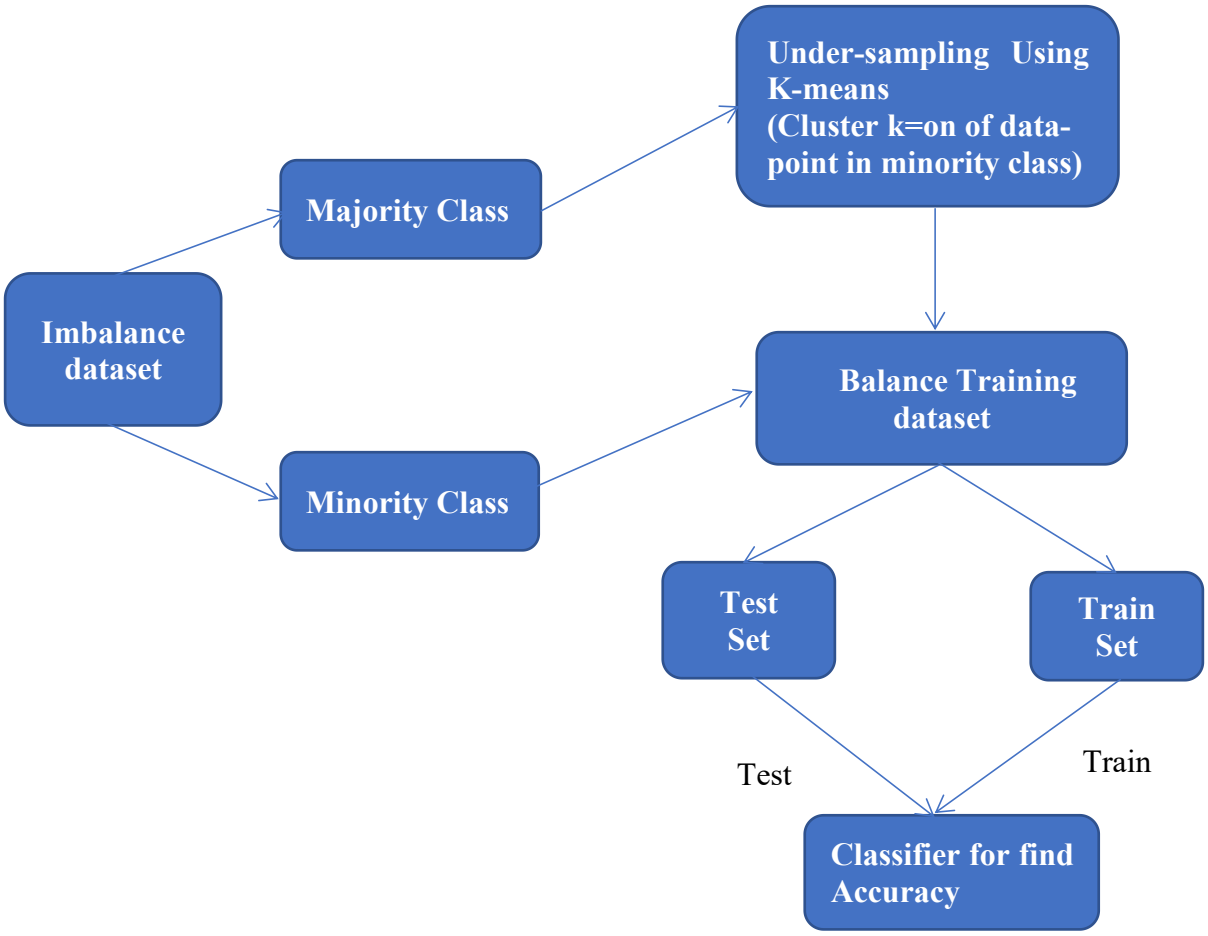

Figure 3.2: Proposed Model for Clustering-based Under-sampling

- The first step is to divide the training set into a majority class subset and a minority class subset.
- The second step is the K-means algorithm where the number of cluster K is equal to the number of data-point in minority class N is employed to reduce the number of data samples in the majority class.
- The third step to take the nearest neighbor of the centroid point so the number of data-point in the majority class is equal to the number of data-point in the minority class.
- The fourth step to reduced the majority class subset is then combined with the minority class subset, resulting in a balanced training set.
- The fifth step is to divide this imbalanced data set into training and testing sets based on the *k*-fold cross-validation method.
- Finally, the classifier is trained and tested by the balanced training and testing sets, respectively.

The respective flow diagram is given below for better understanding. For the classifier, this experiment has used Support Vector Machine. SVM is a supervised learning algorithm that is briefly discussed in the previous chapter.

3.3 Conclusion

The proposed model works better than the previous model. The analytical analysis will be discussed in the next chapter. This chapter discusses the difference and the comparison of the two models.

Chapter 4

Experimental Results and Performance Analysis

This chapter consists of the results and performance analysis of the research. It also analysis the data source that we worked with. Later in the chapter, the results are compared with other existing work on this topic.

4.1 Data Set

The dataset consists of protein examples. Each line describes one example. The structure of each line is as follows

- The first element of each row is BLOCK ID that defines to which native sequence that example belongs. There is a unique BLOCK ID for each sequence. BLOCK IDs are integers running from 1 to 303. One for each native sequence.
- The second element of each row is an EXAMPLE ID that uniquely describes the example.
- The third element of the class of the example. Proteins that are homologous to the native sequence are denoted by 1.Non-homologous proteins by 0.
- All following elements are feature values. There are 74 feature values in each row. The feature describes the match between the native protein sequence and the sequence that is tested for homologous.
- There are no missing values in the protein dataset.

Example of data

279 261532 0 52.00 3269 …….. -0.350 0.26 0.76

Here 279 is the BLOCK ID. 261532 is EXAMPLE ID. The "0" in the third column is the target value. This defines that this protein is not homologous to the native sequence. If this target value is "1" then the protein sequence would be homologous. Columns 4 to 77 are the input attributes. The elements in each line are separated by whitespace.Table-1 contains dataset information. Here the total number of features is 74 and the number of a datapoint is 145,751. The number of data points in the majority class is 144,455 and the number of data points in the minority class is 1296. So the imbalance ratio is 111.46 which is much high.

Table-1:dataset information

Name of dataset	No of the data samples	No of features	No of data in the majority class	No of data in the minority class	Imbalance ratio
Protein homology prediction	**145,751**	**74**	**144455**	**1296**	**111.46**

In figure 4.1 show that the number of datapoint in the majority class and the number of minority class data point. Here Blue bar represents the majority class whose value is 144,455 and the orange bar represents the minority class whose value is 1296. The imbalanced ratio is 111.46. The orange bar er much smaller because of its small data point.

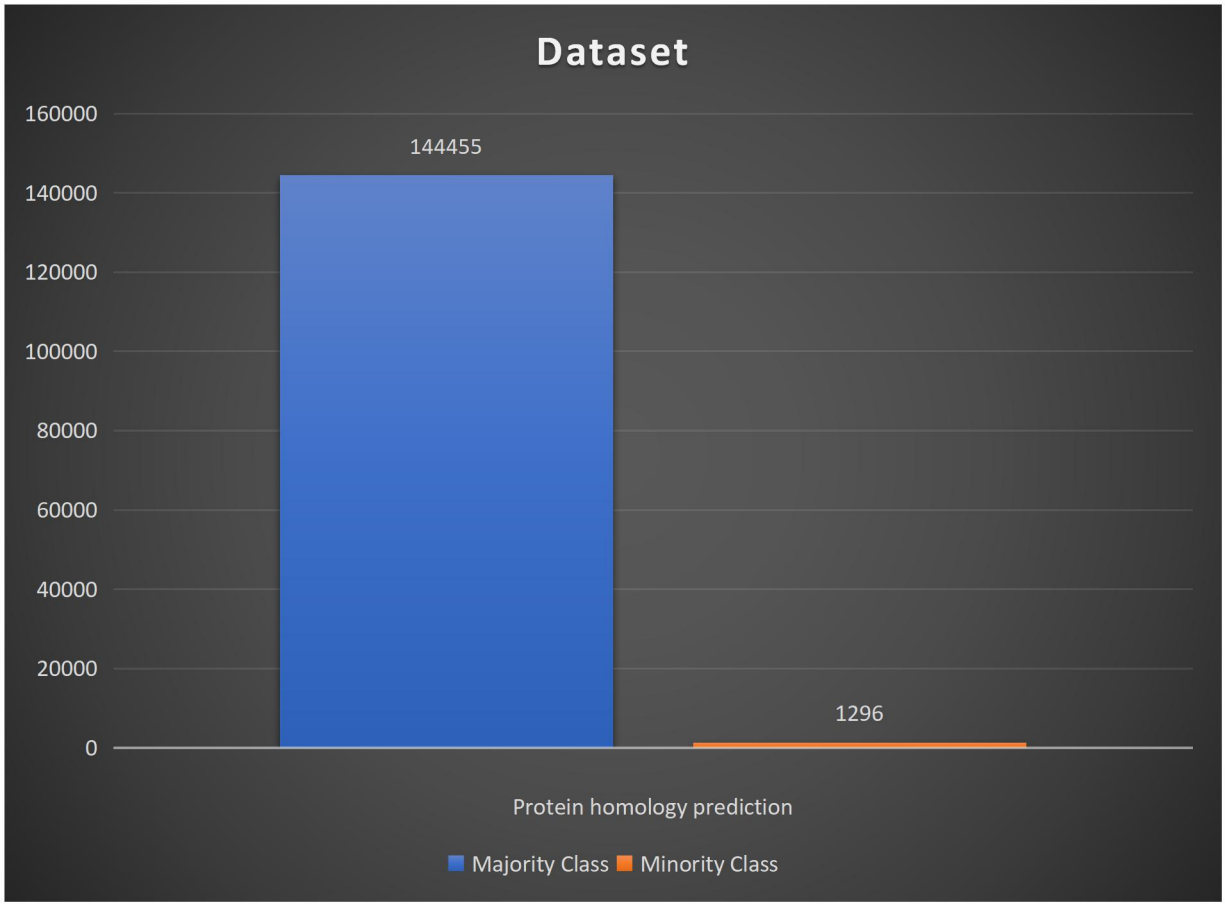

Figure 4.1:Protein Homologous Prediction dataset

4.2 Procedure

The steps for the experiment are described in the methodology chapter. Steps are shortly given here

- The first step is to divide the training set into a majority class subset and a minority class subset.
- The second step is the K-means algorithm where the number of cluster K is equal to the number of data-point in minority class N is employed to reduce the number of data samples in the majority class.
- The third step to take the nearest neighbor of the centroid point so the number of data-point in the majority class is equal to the number of data-point in the minority class.

- The fourth step to reduced the majority class subset is then combined with the minority class subset, resulting in a balanced training set.
- The fifth step is to divide this imbalanced data set into training and testing sets based on the *k*-fold cross-validation method.
- Finally, the classifier is trained and tested by the balanced training and testing sets, respectively.

4.3 Performance Evolution Matrix

In figure 4.2 have shown the confusion matrix. Here

TP= True Positive (Interpretation: If someone predicted positive and it's True)

TN= True Negative(Interpretation: If someone predicted Negative and it's True)

FP= False Positive(Interpretation: If someone predicted positive and it's False)

FN= False Negative(Interpretation: If someone predicted Negative and it's False)

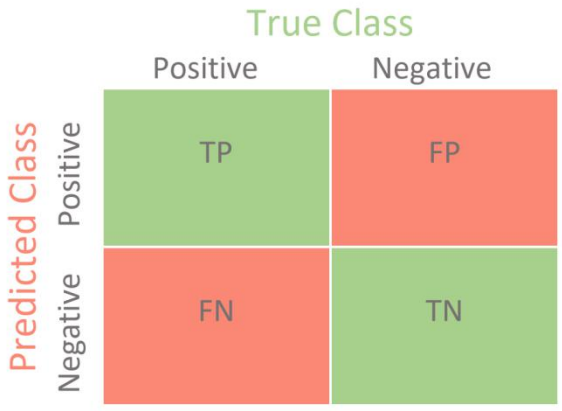

Figure 4.2:Confusion Matrix

Accuracy: It gives us the overall accuracy of the model, meaning the fraction of the total samples that were correctly classified by the classifier.
The equation for calculation of accuracy is

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

Misclassification Rate: It tells us what fraction of predictions were incorrect. It is also known as Classification Error. The equation for calculation of misclassification rate is

$$Rate = \frac{\mathrm{FP} + \mathrm{FN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

Precision: It tells us what fraction of predictions as a positive class were actually positive. The equation for calculation of precision is

$$precision = \frac{TP}{TP + FP}$$

Recall: It tells us what fraction of all positive samples were correctly predicted as positive by the classifier. It is also known as True Positive Rate (TPR), Sensitivity, Probability of Detection. The equation for calculation of recall is.

$$\mathrm{Re}\,call = \frac{TP}{TP + FN}$$

Specificity: It tells us what fraction of all negative samples is correctly predicted as negative by the classifier. It is also known as True Negative Rate (TNR). The equation for calculation of specificity is

$$Specificity = \frac{TN}{TN + FP}$$

F1-score: It combines precision and Recall into a single measure. Mathematically it's the harmonic mean of precision and recall. The equation for calculation of F1-score is

$$F_1 - score = \frac{2TP}{2TP + FP + FN}$$

4.4 Experimental Result

The Accuracy, Misclassification Rate, Precision, Recall, Specificity, and F1-score after using K-means in the majority class and balance the dataset are shown in Table-2.

Table-2 consist of all the performance evolution matrix values. The first column consists of the SVM classifier and the different number of cluster values. K=N is the balanced dataset where K is equal to the number of clusters in the majority class and N is equal to the number of datapoint in the minority class. Then

$$K = N \pm 5$$
$$K = N \pm 10$$

Varies the number of clusters and evolute the performance evolution.

Table-2:Performance evolution of Proposed Model

SVM Classifier	Accuracy	Miss classification Rate	Precision	Recall	Specificity	F1-score
K=N	0.965	0.035	0.953	0.976	0.977	0.964
K=N-5	0.955	0.044	0.969	0.944	0.941	0.957
K=N-10	0.948	0.051	0.935	0.962	0.963	0.948
K=N+5	0.945	0.055	0.953	0.936	0.938	0.945
K=N+10	0.958	0.041	0.961	0.952	0.956	0.956

Here, when K=N then the Accuracy, Recall, Specificity, F1-score are maximum, and when K=N+5 misclassification rate is maximum and when K=N-5, Precision is maximum. And when K=N then the misclassification rate is minimum which is our goal.

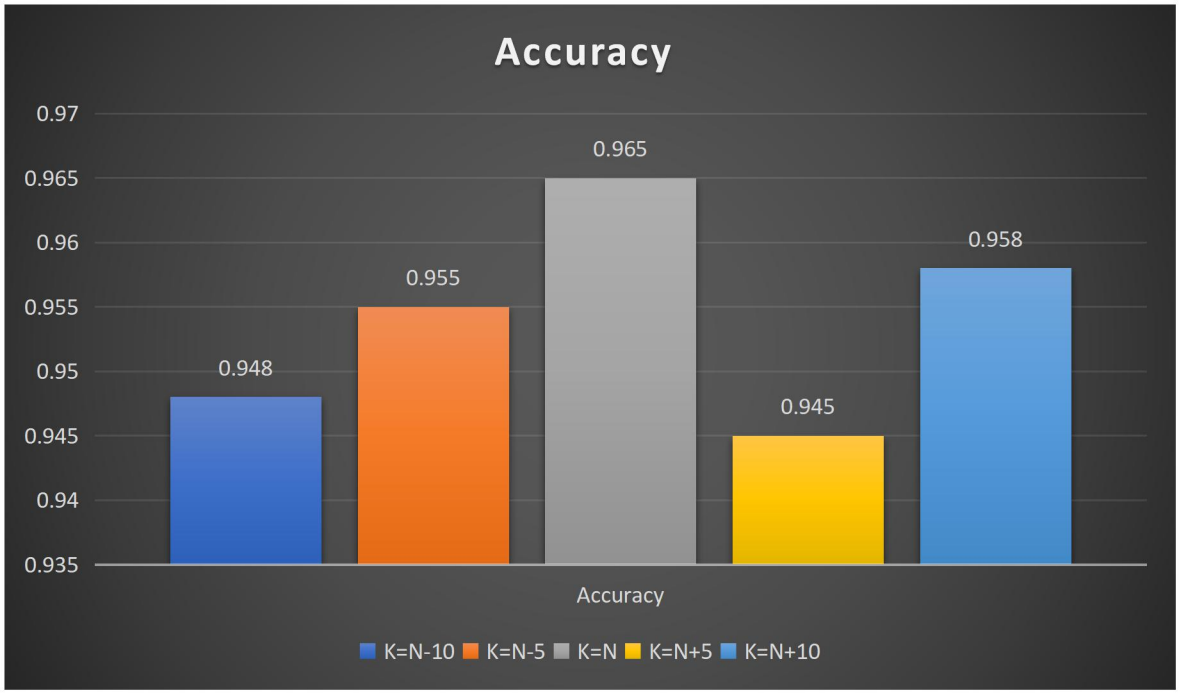

Figure 4.3:Accuracy of Proposed model

In figure 4.3 have shown the accuracy of the proposed model. Here when K=N means when the number of clusters in the majority class is equal to the number of datapoint in the minority class then the accuracy is maximum(0.965). And when K=N+5 means the number of clusters is greater than the number of datapoint in minority class accuracy is minimum(0.945).

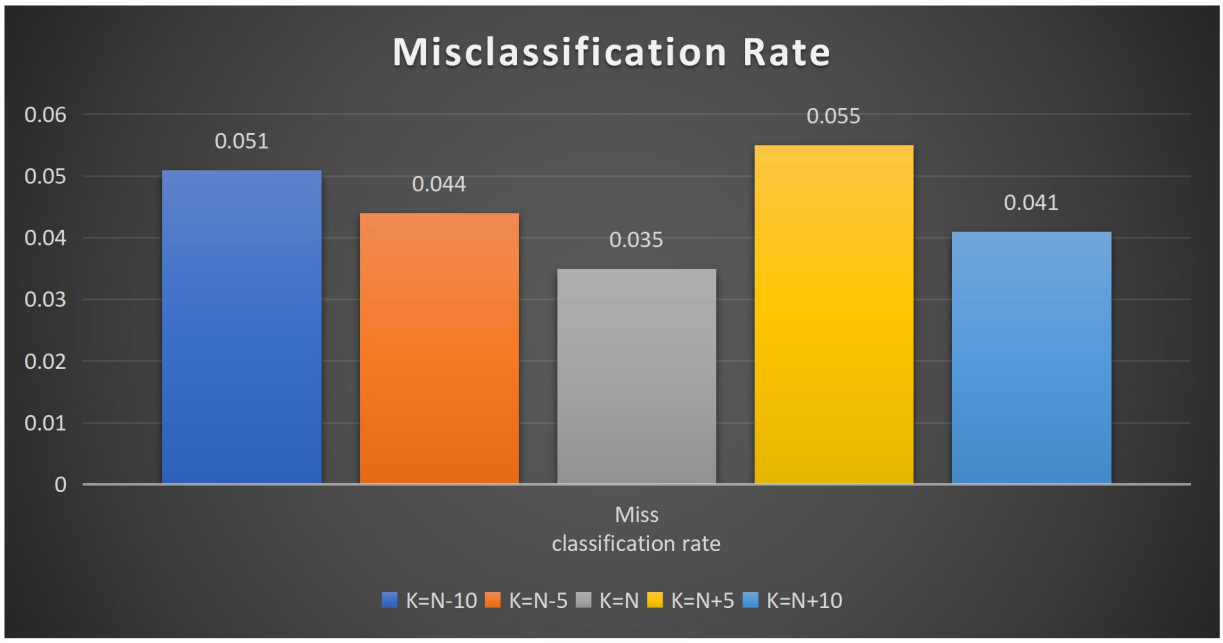

Figure 4.4:Misclassification Rate of Proposed

In figure 4.4 have shown the Misclassification Rate of the proposed model. Here when K=N Misclassification Rate is minimum(0.035) and when K=N+5 Misclassification Rate is maximum(0.055). Our goal is to reduce the Misclassification Rate. So it is reduced when K=N.

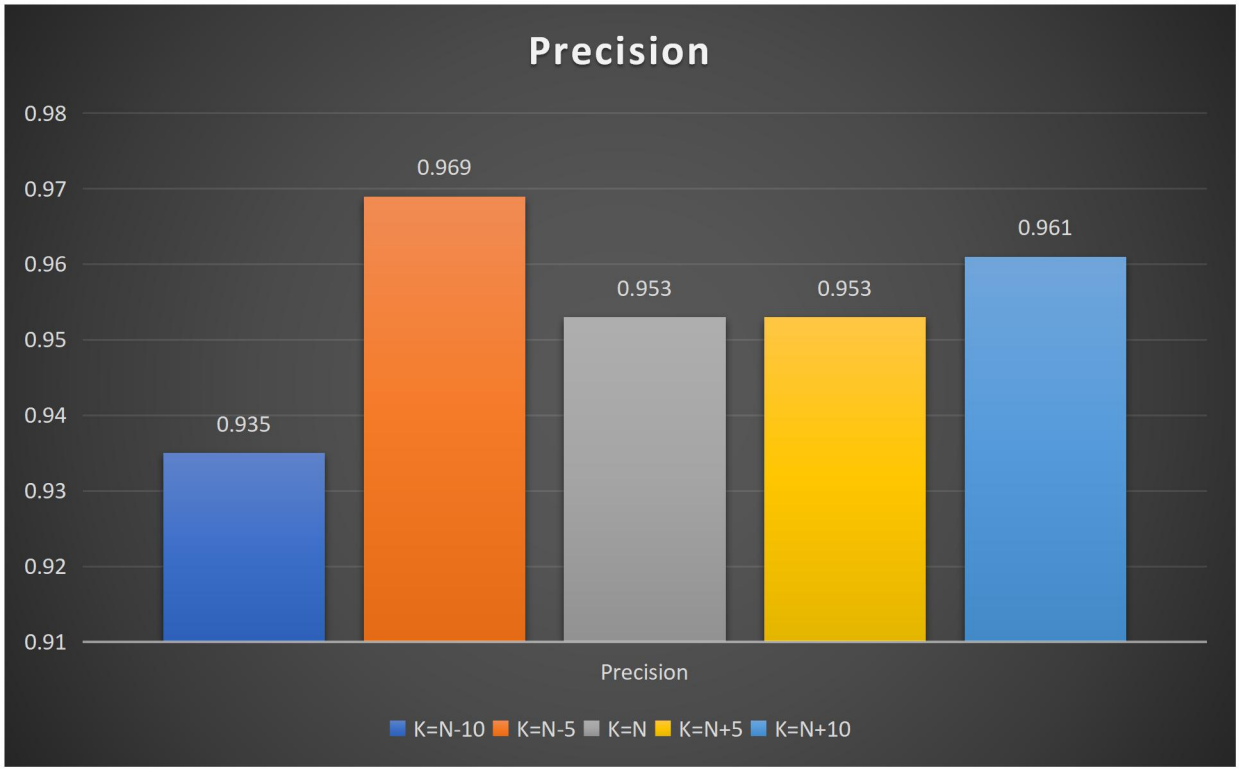

Figure 4.5:Precision of Proposed model

In figure 4.5 have shown the precision value of the proposed model. Here when K=N-5 means the number of clusters is less than the number of datapoint in minority class got maximum(0.969) precision value and when K=N-10 then the precision value is minimum(0.935).

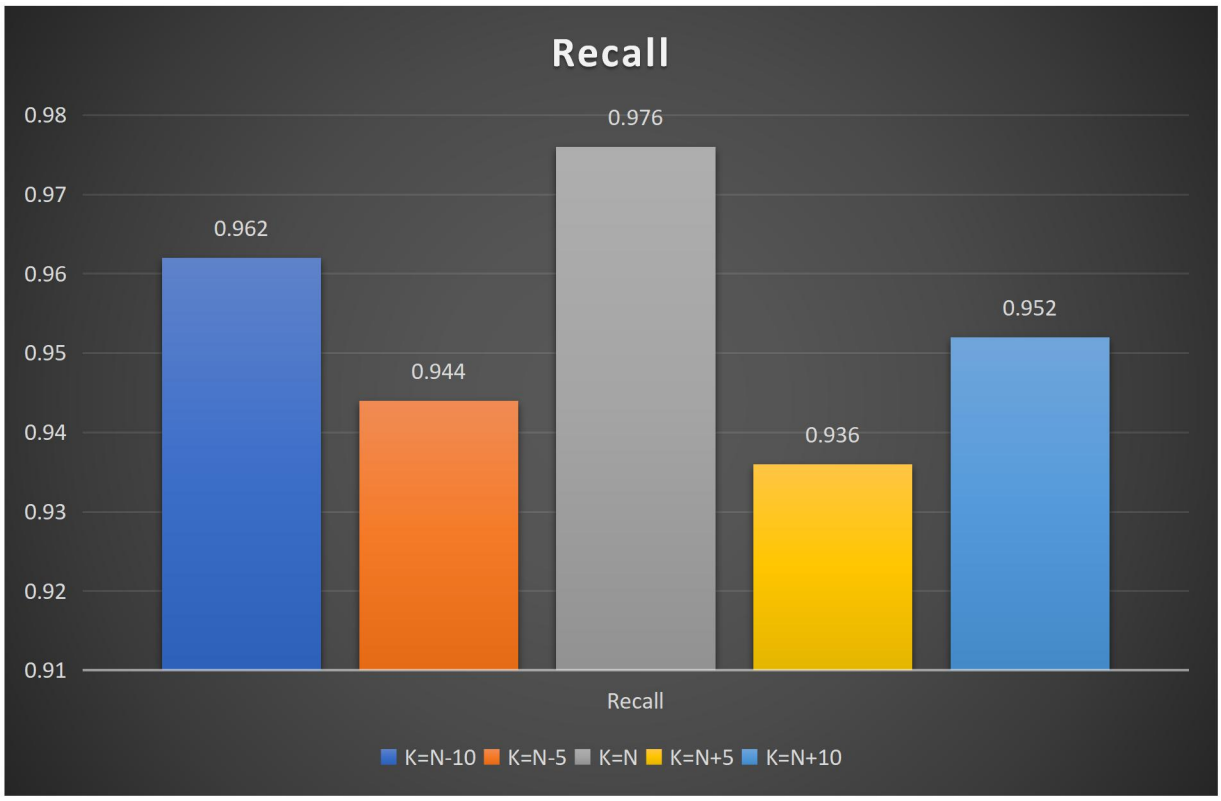

Figure 4.6:Recall of Proposed model

In figure 4.6 have shown the Recall value of the proposed model. Here When K=N, the Recall value is maximum(0.976), and when K=N+5, the Recall value is minimum (0.936). So when the number of clusters is equal to the number of datapoint in the minority class then recall is maximum which is our target.

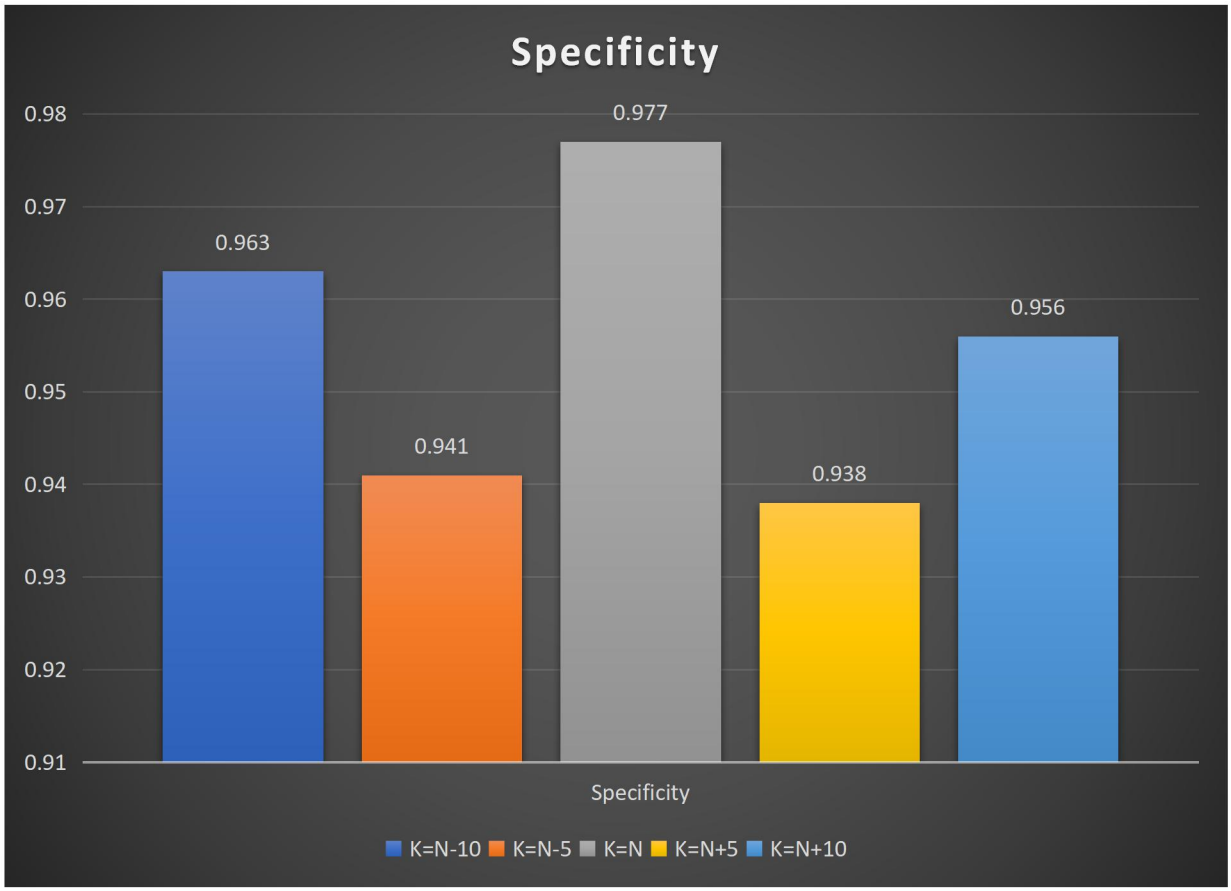

Figure 4.7:Specificity of Proposed model

In figure 4.7 have shown the Specificity value of the proposed model. Here When K=N, the Specificity value is maximum(0.977), and when K=N+5, the specificity value is minimum (0.938). So when the number of clusters is equal to the number of datapoint in minority class then Specificity is maximum which is our target.

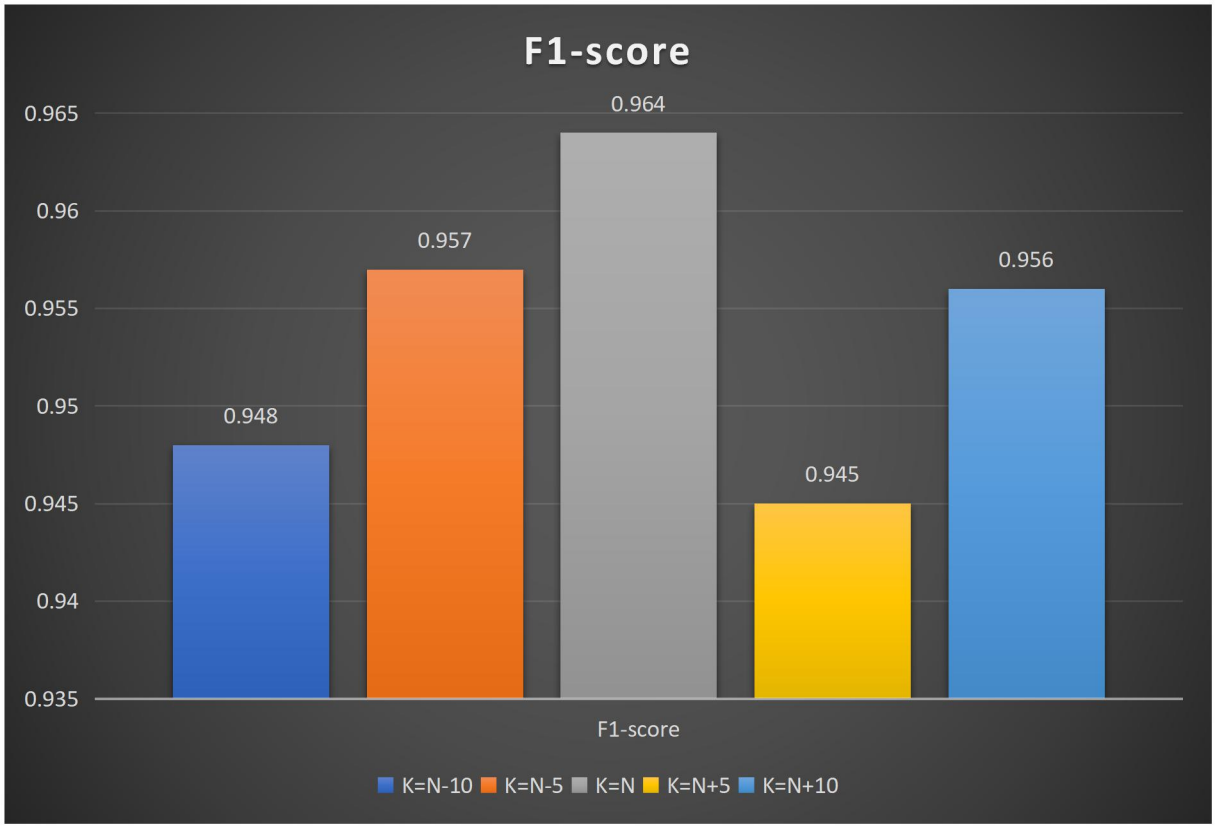

Figure 4.8:F1-score of Proposed model

In figure 4.8 have shown the F1-score value of the proposed model. Here When K=N, the F1-Score value is maximum(0.964), and when K=N+5, the F1-Score value is minimum (0.945). So when the number of clusters is equal to the number of datapoint in minority class then F1-score is maximum which is our target.

4.5 Comparison

The main difference between the previous model and the proposed model is that the previous model conduct all the calculations by using WEKA software and the proposed model provides raw python code. The basic idea about WEKA is given below

4.5.1 Result Comparison

Comparison between the result is given in the below table

Table-3: Competitive Accuracy between two models

SVM Classifier	K=N	K=N-5	K=N-10	K=N+5	K=N+10
Previous Model	0.755	0.697	0.669	0.715	0.696
Proposed Model	0.965	0.955	0.948	0.945	0.958

Table-3 have shown the previous model and proposed model accuracy. Here in the previous model when K=N, accuracy is maximum, and when K=N+10, accuracy is minimum. And in the proposed model when K=N, accuracy is maximum, and when K=N+5, accuracy is minimum. So from this table, the should take that when K=N means the number of clusters is equal to the number of datapoint in minority class, then the accuracy is maximum always. And when the number of clusters is greater than the number of datapoint in the minority class, then the accuracy is always less.

The competitive result is shown graphically in figure 4.9.

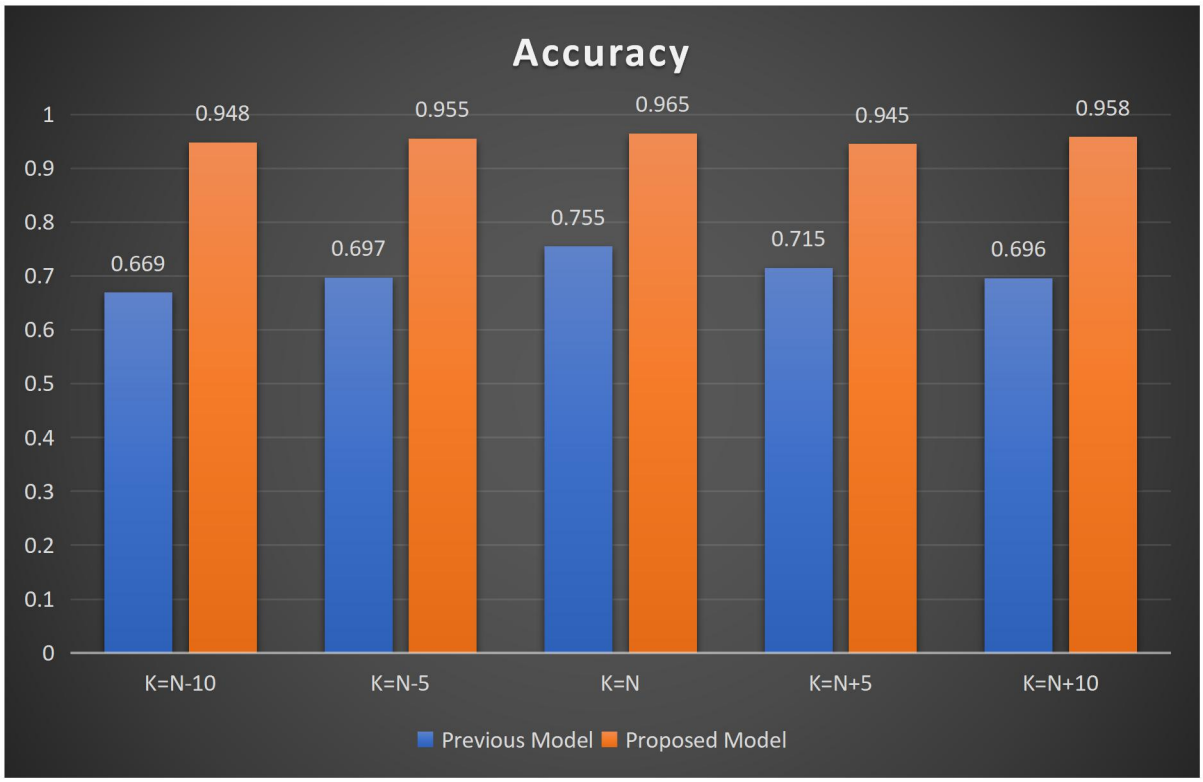

Figure 4.9: Accuracy difference between the previous and proposed

In figure 4.9 have shown the comparison side by side. Here the blue bars define the previous model and the orange bars define the proposed model. The relation between the previous model and the proposed model is proportional

4.6 Conclusion

This chapter covers the competitive result between the previous model and the proposed model. The accuracy of the proposed model is much higher than the previous model.

CHAPTER 5

Conclusion and Future Works

This chapter discusses the summary of the thesis work, its limitation, and shows future work direction.

5.1 Summary

Class Imbalance problem is a very common real-world problem. It is very difficult to collect a balanced dataset in the real world. So it should be controlled at the data level or algorithm level. At the data level, it's much easier to handle the imbalanced problem rather than the algorithm level. In the data level solution, this experiment follow those step given below

First divided the whole dataset into majority and minority classes. Then applied K-means in the majority class where the number of clusters is equal to the number of datapoint in the minority class. Then took the nearest neighbor of the centroid. After that balanced the dataset with the minority class. After Splitting the dataset into 60% as a train set and 40% as a test set. Then applied SVM for train and Test and calculate the accuracy. Lastly Compared the calculated accuracy with the previous model accuracy.

5.2 Application

The class imbalance problem is a common real-world problem. In the real world, it is very difficult to collect a balanced dataset. In real life, various fields will cover if the class imbalance problem is solved. In real life, the given field problem can be solved by solving the class imbalance problem.

Fraud Detection, Claim Prediction, Default Prediction, Churn Prediction, Spam Detection, Anomaly Detection, Outlier Detection, Intrusion Detection, Conversion Prediction are the real-world problem which can be solved by solving the class imbalance problem. So solving the class imbalance problem has a real-world effect.

5.3 Limitations

We worked with only one data set to train and test our model. Also used only one classifier for train and test. So if we work with various classifiers and different types e datasets then the result may change.

5.4 Future Work

The plan is to keep working with the new imbalance dataset and also applied different types of the classifier. The plan includes:

- We want to work with other different kinds of short or big imbalanced datasets.
- We also want to develop:
    1. K-NN classifier for train and test for imbalanced dataset.
    2. MLP classifier for train and test for imbalanced dataset.

5.5 Conclusion

The work was completed without facing a major problem. It was a success in handling the imbalanced dataset for the classification problem.

REFERENCES

[1] Lemaître, Guillaume, Fernando Nogueira, and Christos K. Aridas. "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning." *The Journal of Machine Learning Research* 18.1 (2017): 559-563.

[2] García, Vicente, et al. "Combined effects of class imbalance and class overlap on instance-based classification." *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, Berlin, Heidelberg, 2006.

[3] Ali, Aida, Siti Mariyam Shamsuddin, and Anca L. Ralescu. "Classification with class imbalance problem: a review." *Int. J. Advance Soft Compu. Appl* 7.3 (2015): 176-204.

[4] Lin, Wei-Chao, et al. "Clustering-based undersampling in class-imbalanced data." Information Sciences 409 (2017): 17-26.

[5] Yen, Show-Jane, and Yue-Shi Lee. "Cluster-based under-sampling approaches for imbalanced data distributions." *Expert Systems with Applications* 36.3 (2009): 5718-5727.

[6] Tao, Yanyun, Yuzhen Zhang, and Bin Jiang. "DBCSMOTE: a clustering-based oversampling technique for data-imbalanced warfarin dose prediction." *BMC medical genomics* 13.10 (2020): 1-13.

[7] Rekha, Gillala, V. Krishna Reddy, and Amit Kumar Tyagi. "A novel approach for solving skewed classification problem using cluster based ensemble method." *Mathematical Foundations of Computing* 3.1 (2020): 1.

[8] Guo, Xinjian, et al. "On the class imbalance problem." *2008 Fourth international conference on natural computation*. Vol. 4. IEEE, 2008.

[9] Tsai, Chih-Fong, et al. "Under-sampling class imbalanced datasets by combining clustering analysis and instance selection." *Information Sciences* 477 (2019): 47-54.

[10] Kumar, N. Santhosh, et al. "Undersampled $$$$-means approach for handling imbalanced distributed data." *Progress in Artificial Intelligence* 3.1 (2014): 29-38.

[11] Ofek, Nir, et al. "Fast-CBUS: A fast clustering-based undersampling method for addressing the class imbalance problem." *Neurocomputing* 243 (2017): 88-102.

[12] Sobhani, Parinaz, Herna Viktor, and Stan Matwin. "Learning from imbalanced data using ensemble methods and cluster-based undersampling." *International Workshop on New Frontiers in Mining Complex Patterns*. Springer, Cham, 2014.

[13] Onan, Aytuğ. "Consensus clustering-based undersampling approach to imbalanced learning." *Scientific Programming* 2019 (2019).

[14] Zhang, Jue, and Li Chen. "Clustering-based undersampling with random over sampling examples and support vector machine for imbalanced classification of breast cancer diagnosis." *Computer Assisted Surgery* 24.sup2 (2019): 62-72.

[15] Altınçay, Hakan, and Cem Ergün. "Clustering based under-sampling for improving speaker verification decisions using AdaBoost." *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, Berlin, Heidelberg, 2004.

[16] Chawla, Nitesh V. "Data mining for imbalanced datasets: An overview." *Data mining and knowledge discovery handbook*. Springer, Boston, MA, 2009. 875-886.

[17] Kotsiantis, Sotiris, Dimitris Kanellopoulos, and Panayiotis Pintelas. "Handling imbalanced datasets: A review." *GESTS International Transactions on Computer Science and Engineering* 30.1 (2006): 25-36.

[18] Akbani, Rehan, Stephen Kwek, and Nathalie Japkowicz. "Applying support vector machines to imbalanced datasets." *European conference on machine learning*. Springer, Berlin, Heidelberg, 2004.

[19] Ganganwar, Vaishali. "An overview of classification algorithms for imbalanced datasets." *International Journal of Emerging Technology and Advanced Engineering* 2.4 (2012): 42-47.

[20] Ramyachitra, D., and P. Manikandan. "Imbalanced dataset classification and solutions: a review." *International Journal of Computing and Business Research (IJCBR)* 5.4 (2014).