2018

Application of Synthetic Informative Minority Over-Sampling (SIMO) Algorithm Leveraging Support Vector Machine (SVM) On Small Datasets with Class Imbalance

Akshatha Fakkeriah Kallappanamatt
Technological University Dublin, Ireland

Application of Synthetic Informative Minority Over-Sampling (SIMO) Algorithm Leveraging Support Vector Machine (SVM) On Small Datasets with Class Imbalance

Akshatha Fakkeriah Kallappanamatt

A dissertation submitted in partial fulfilment of the requirements of
Dublin Institute of Technology for the degree of
M.Sc. in Computing (Data Analytics)

September - 2018

DECLARATION

I certify that this dissertation which I now submit for examination for the award of MSc in Computing (Data Analytics), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the test of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Dublin Institute of Technology and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the Institute's guidelines for ethics in research.

Signed: Akshatha Fakkeriah Kallappanamatt

Date: 04 September 2018

ABSTRACT

Developing predictive models for classification problems considering imbalanced datasets is one of the basic difficulties in data mining and decision-analytics. A classifier's performance will decline dramatically when applied to an imbalanced dataset. Standard classifiers such as logistic regression, Support Vector Machine (SVM) are appropriate for balanced training sets whereas provides suboptimal classification results when used on unbalanced dataset. Performance metric with prediction accuracy encourages a bias towards the majority class, while the rare instances remain unknown though the model contributes a high overall precision. There are chances where minority instances might be treated as noise and vice versa. (Haixiang et al., 2017). Wide range of Class Imbalanced learning techniques are introduced to overcome the above-mentioned problems, although each has some advantages and shortcomings.

This paper provides details on the behavior of a novel imbalanced learning technique Synthetic Informative Minority Over-Sampling (SIMO) Algorithm Leveraging Support Vector Machine (SVM) on small datasets of records less than 200. Base classifiers, Logistic regression and SVM is used to validate the impact of SIMO on classifier's performance in terms of metrices G-mean and Area Under Curve. A Comparison is derived between SIMO and other algorithms SMOTE, Smote-Borderline, ADAYSN to evaluate performance of SIMO over others.

Key words: Class imbalance, Class imbalance learning, Machine learning, Supervised learning, Small datasets.

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to the supervisor Dr. Luca Longo for his encouragement and support, imparting constructive suggestions and recommendations and guiding me throughout the research project.

I would like to express my regards to one of the author of SIMO algorithm, Professor Saeed Piri for clarifying algorithm related queries when necessary. I would like to thank my classmate Jaydeep for reviewing and helping in translating the algorithm from Matlab code to python.

I would also like to express my gratitude towards all the ‘Dublin Institute of Technology academic staffs’ especially the professors who shared valuable ken and for their help, support and guidance throughout the academic.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

LIST OF ACRONYMS

AUC	Area Under the Curve
CRISP-DM	Cross Industry Standard Process for Data Mining
CV	Cross Validation
FN	False Negative
FP	False Positive
TN	True Negative
TP	True Positive
TPR	True Positive Rate
FPR	False Positive Rate
SMOTE	Synthetic Minority Over-Sampling Technique
SVM	Support Vector Machine
TN	True Negative
KNN	K Nearest Neighbour
SVM	Support Vector Machine
ROC	Receiver Operating Characteristic
SIMO	Synthetic informative minority Oversampling
MSMOTE	Modified Synthetic Minority Oversampling
ADASYN	Adaptive Synthetic Sampling
ASUWO	Adaptive semi-unsupervised weighted oversampling
SUNDO	Similarity-based Under Sampling and Normal Distribution-based Oversampling
SMOTE-B1	SMOTE Borderline 1
SMOTE-B2	SMOTE Borderline 2

CHAPTER 1

INTRODUCTION

1.1 Background

Health informatics is defined as “all aspects of understanding and promoting the effective organization, analysis, management, and use of information in health care”. Whereas, Data mining is defined as “a process of nontrivial extraction of implicit, previously unknown and potentially useful information from the data stored in a database”, the core step in the Knowledge discovery in databases (KDD). It is the nontrivial process identifying valid, novel, potentially useful, and ultimately understandable patterns in data. The amalgamation of these two is becoming popular nowadays because with the help of an appropriate computer-based systems and efficient analytical methodologies one can meticulously discover the significant hidden knowledge from huge medical databases which includes finding correlations or patterns among different fields in large medical databases. (Kavakiotis et al., 2018)

Predictive Analytics is nothing but the application of data mining techniques incorporating machine learning algorithms on historic data in predicting the future or the unknown. It is gaining a wide range of importance in various disciplines and healthcare domain is one among them. Machine learning can be formally defined as, A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E. Application of predictive analytics in healthcare domain majorly includes, Prediction of presence or susceptibility of an individual to disease, mortality risks considering different scenarios, survivability of a patient after a medical treatment and so on. (Kavakiotis et al., 2018)

Machine learning is classified into three categories namely Supervised learning, Unsupervised learning and Reinforcement learning. Unsupervised learning draw inferences from unlabeled datasets whereas Supervised learning predicts the unknown with a prior knowledge of the datasets. It uses labelled data. Classification is one of the supervised

learning method which is used to predict the target set which is dichotomous in nature. This can be achieved by available algorithms like Logistic regression, Super Vector Machine, Decision tree, Random forest, etc. (Sharma & Sharma, 2016)

The usual problem faced by classification algorithms that would make its functionality worthless is Class imbalance problem. Standard Classifiers like Logistic regression and SVM works fine on balanced datasets but their performance deteriorates when it comes to working on Imbalanced datasets. Class imbalance is a scenario where the examples of one class significantly outnumbers the other and the branch of machine learning which deals with such problems is named Class Imbalance learning. Recent researches are more focused on imbalanced and overlapping datasets as real data is often skewed like medical diagnosis, oil blowout detection, financial fraud detection, network intrusion detection, spam detection, text classification, etc. Problem of small disjuncts and small sample size with high feature dimensionality causing classification errors are often encountered and these are closely related problems to class imbalance problem.

Imbalance learning algorithm can be categorized into data-level approach and algorithm-level approach. The data-level approach occurs at the pre-processing phase whereas algorithm level is modifying the learning algorithms itself to perform efficiently on imbalanced data. Data-level approach includes Resampling which is again categorized into Oversampling and Under sampling techniques. Oversampling technique is a popular approach especially oversampling by synthetic data generation has gained a huge research importance, popular ones are SMOTE, Borderline-SMOTE, ADASYN. Many approaches have been proposed having its own merits and demerits wherein Oversampling by Synthetic data generation.

The problem of overfitting, overgeneralization of the models, chance of oversampling noise examples which will increase the misclassification rate are few drawbacks which most of the oversampling algorithms come across. To overcome these disadvantages Cost sensitive and Ensemble-based learning techniques are approached. Cost-sensitive methods assigns different costs to the samples of different classes to make minority examples more important during training process. Ensemble learning methods is the combination of ensemble learning algorithms and any of the above-mentioned

approaches. This includes, SMOTEBoost, SMOTEBagging, AdaCost and so on. (Peng, 2015).

1.2 Research Project

The objective of the experiment is to evaluate the performance of novel imbalanced learning technique, Synthetic Informative Minority Over-Sampling (SIMO) Algorithm Leveraging Support Vector Machine (SVM) on small datasets. The performance comparison is carried out with existing imbalance learning techniques like SMOTE, SMOTE-Borderline and ADASYN which are considered as baseline algorithms for this research. Standard Supervised learning algorithms Linear algorithms and SVM is used to assess the impact of SIMO on these classifier's prediction performance. To obtain optimized results from the classifiers GridsearchCV with five folds is used. Since the experiment involves small datasets of class imbalance problem, performance metrices G mean and Area Under Curve is considered for the evaluation and accuracy is prioritized the least. Five iterations are carried out and the mean G mean and AUC is tabulated to obtain visual representations for easier analysis.

The research question which can be answered from this research is as stated below,

"*Can performance of the classifiers on small datasets, significantly improve on the application of 'SIMO leveraging SVM' over the application of baseline imbalanced learning algorithms?*"

***SIMO: Synthetic Informative Minority Over-Sampling*

***SVM: Support Vector Machine*

***Baseline imbalanced learning algorithms: SMOTE, SMOTE-Borderline1, SMOTE- Borderline2 and ADASYN.*

***Performance metrices: Accuracy, G-mean and AUC*

1.3 Research Objectives

The key objective of this research is to evaluate the performance of novel imbalanced learning technique, Synthetic Informative Minority Over-Sampling (SIMO) Algorithm Leveraging Support Vector Machine (SVM) on small datasets. To achieve the same four datasets are chosen which belongs to univariate classification problem and following steps are carried out,

1. Data understanding comprising a detailed analysis of the datasets in terms of its distribution through graphs, minimum and maximum values, central tendency and standard deviation of the variables and the relationship they share with the target and themselves.
2. Data cleaning involves removal of missing values and imputation.
3. Data preparation is carried out which includes Standardization (Z-scores), One-hot Encoding.
4. Data partition with a stratified split of 75 percent train data and 25 percent test data.
5. Application of Oversampling techniques SMOTE, SMOTE-Borderline1, SMOTE-Borderline2, ADASYN and SIMO on train data.
6. Resampled or oversampled data is trained using supervised learning algorithms Logistic Regression and SVM.
7. The model results are recorded with respect to G mean, AUC obtained from ROC plots and Accuracy from confusion matrix.
8. Graphical representation is obtained for model's performance comparison on the application of imbalanced learning algorithms.
9. Use of Wilcoxon Signed rank test to statistically asses the results and determine the rejection or acceptance of research hypothesis that answers research question.

1.4 Research Methodologies

The key focus of this research is to evaluate the impact of novel imbalanced learning technique, Synthetic Informative Minority Over-Sampling (SIMO) Algorithm Leveraging Support Vector Machine (SVM) on classifiers performances for small datasets. Hence, existing datasets which are small in sample size and have moderate to high class imbalance are chosen.

The type of research carried out is Secondary and the methodology involves a systematic empirical investigation of quantitative properties available in the collected information. The results obtained from the experiment will be used as a source of support in rejecting or retaining the hypothesis that will in turn answer the research question.

1.5 Scope and Limitations

The scope of the project is to examine the influence of novel imbalanced learning technique, Synthetic Informative Minority Over-Sampling (SIMO) Algorithm Leveraging Support Vector Machine (SVM) on small datasets. Usually domains like Biomedical and their related fields, health informatics will have small data to analyze and classifiers usually will fail to perform well as there will be no sufficient information available to learn. Small datasets with class imbalance can make classifier even worthless as they tend to misclassify minority examples as majority since majority examples are present in abundance. The current study is focused on similar problem investigating the applicational effect of SIMO on four datasets with small sample size with class imbalance.

Considering only four datasets to evaluate SIMO algorithm is the limitation of this research. Understanding and pre-processing of the datasets of records less than 150 without losing information and treating the class imbalance associated with it is another limitation to overcome.

1.5 Document Outline

The Report document includes following section and the contents covered in each section is descripted below.

Chapter 2 (Literature Review) gives an overview of the literatures related to Class Imbalance problem, its impact on classifiers and various learning methods proposed to overcome the limitations. This section also discusses benefits and shortcomings of the proposed methods and how one is efficient than other. It also explains the working of SIMO algorithm, its merits and demerits in detail.

Chapter 3 (Design and Methodology) outlines the design implemented in the research, techniques involved in the design and its purpose, its advantage and limitations, usefulness of its implementation.

Chapter 4 (Results and Discussions) provides an account of results obtained on implementing the proposed design in the previous section. It provides a detailed discussion on the results obtained and provides comparison of model's performance based on achieved results. Section also discusses the difference in expected and actual result of the experiment, difference in the observations found from current experiment with original literature.

Chapter 5 (Conclusion) summarizes the research carried out, approaches used in the implementation and results obtained on the same. Contribution and the impact of proposed research is also accounted in this chapter. Furthermore, it discuses about the future work and recommendations.

CHAPTER 2

LITERATURE REVIEW

This chapter provides a review on the literatures available on imbalanced learning algorithms. This includes the definition of Class imbalance, effect of imbalanced datasets on learners, different methods involved in balancing a dataset and its performance and advantage over other methods. The section also explains the gap in the research which serves as a motive for this experiment.

2.1 Class Imbalance

Class imbalance is commonly found in classification problem field where a class examples significantly outnumbers the other and is not equally represented. The real-world data will always have imbalanced class distribution. There will be a substantial loss of performance due to skewed class distribution which is determined by imbalance ratio. Imbalance ratio is the ratio of majority class instances to the minority class instances. The level of imbalance could be as huge as10^6.

The reliability of the model is dependent on quality of training data and hence it should be representative and should be informative for the learners. Training data with imbalanced class problem will significantly degrade the performance of the model with longer computational timing. Class imbalance can be usually found in medical diagnosis, financial fraud detection, spam detection, text classification and so on. (Mi, 2013)

2.2 Effect of Class imbalance

The problems which are usually faced while dealing with imbalanced datasets are lack of density, data shift, problem of overlapping, identification of noisy examples available at the borderline and its effect. In a classification problem, lack of density or information with small dataset will be an issue as learning algorithms will not have enough data to generalize about the distribution of samples which becomes even more difficult with high dimensional imbalanced data. Minority class can be underrepresented, and model used to learn this

dataspace becomes too specific resulting in overfitting and might also induce small disjuncts.

In an empirical study conducted on the effect of imbalance ratio and noise on the classification algorithms and data sampling techniques, it is found that though the classifiers are more sensitive to noise, highly imbalanced data severely hinders the performance of both classifier algorithms and sampling techniques. Thus, it is also important to detect and discriminate between borderline examples and noise instances while sampling. (López, Fernández, García, Palade & Herrera, 2018)

2.3 Machine learning: Class Imbalanced learning technique

In Machine learning, there are many methods proposed to deal with class imbalance problem which can be categorized into data-level approaches and algorithm-level approach, cost-sensitive methods and ensemble of classifiers. Data-level approaches comes into picture while preprocessing the data, to diminish the effect of class imbalance. This includes sampling methods. Algorithm- level approaches create or modify learning algorithms to perform efficiently on class imbalanced datasets. This includes adaptive conformal transformation (ACT) algorithm proposed to change the kernel function of SVM, weighed Euclidean distance function to classify samples using kNN and so on. (Loyola-González, Martínez-Trinidad, Carrasco-Ochoa & García-Borroto, 2018)

Cost-sensitive methods assigns different costs to the samples of different classes to make minority examples more important during training process. Ensemble learning methods is the combination of ensemble learning algorithms and any of the above-mentioned approaches. This includes, SMOTEBoost, SMOTEBagging, AdaCost and so on. (Peng, 2015). This section covers the imbalance learning approach in data-level and hence the relevant and related research studies to the oversampling methods used in this experiment are explained and reviewed in brief.

2.4 Synthetic Data Generation Oversampling Method: Random Minority Samples

Sampling techniques are used to resample the dataset and achieve balanced class distribution. The method of removing the majority class instances to balance the dataset is called under sampling whereas increasing the minority class examples to reduce the degree of imbalanced data distribution is called Oversampling. The basic oversampling technique is random oversampling wherein the minority data points are randomly duplicated, and its major drawback is overfitting. However, oversampling is advisable over under sampling as there will be no chance of losing potentially useful piece of information.

2.4.1 Synthetic Minority Over-Sampling Technique (SMOTE) Technique

This method proposed by (Chawla, Bowyer, Hall & Kegelmeyer, 2002) is an efficient and widely used synthetic data generation oversampling technique in recent days. This method was proposed in place of oversampling with replacement method. Synthetic examples are generated by operating in 'feature space' instead of 'data space. In this approach minority class is oversampled by introducing synthetic data points along the line segments joining k nearest neighbors. Neighbors from the k nearest neighbors are selected based on the oversampling ratio required.

$$X_{new} = X + (X- X') * rand (0,1)$$

where, X' = k nearest neighbor, X = sample

Synthetic samples are generated by multiplying the calculated difference between feature vector and its nearest neighbor with a random number between 0 and 1 and is added back to the feature vector (sample). This results in the selection of a random point along the line segment between two specific features. SMOTE mechanism can be explained as shown in Figure 2.1. The advantage of SMOTE is that it makes the decision regions larger and less specific with a drawback that it generates synthetic instances considering minority examples alone.

2.4.2 Modified Synthetic Minority Oversampling Technique

To improve the performance of SMOTE, a Modified Synthetic Minority Oversampling Technique (MSMOTE) was proposed by (Hu, Liang, Ma & He, 2009). Initially noise from the majority class is removed and the algorithm classifies minority instances into security samples, border samples and latent noise samples by calculating the distance between minority class samples and all the samples of the training dataset. If sample label in minority class is same as labels in k nearest neighbor then sample is security sample, sample is a noise in contrary and sample is borer if it doesn't belong to any of the group. Furthermore, synthetic examples are generated for security samples by randomly choosing one of the k nearest neighbors and on application of this method, the results yielded was better than SMOTE.

2.5 Synthetic Data Generation Oversampling Method: Borderline Minority Samples

The examples on the borderline and the ones nearby are more likely to be misclassified by most of the classification algorithms which attempt to learn the borderline of each class during training process. Thus, the contribution of those examples farther away from the borderline are comparatively less than those which are present on and nearer to borderline. Pointing at these problems, (Han, Wang & Mao, 2005) proposed a technique named Borderline-SMOTE which included *SMOTE-borderline1* and *SMOTE-borderline 2* approaches which are slightly different form each other.

2.5.1 SMOTE- Borderline 1

In this approach, for every instance in the minority class, *m* nearest neighbor is calculated from the whole training dataset. The number of majority examples in the m nearest neighbors is denoted as m'. If all the m nearest neighbors are m', then those examples are noise and is ignored from oversampling. If number of majority neighbors is greater than minority neighbors, then minority instances can be easily misclassified thus these are considered as instances at Danger. The minority instances can be considered safe if majority neighbors are less than minority neighbors. Likewise, the minority data points are categorized into noise, danger and safe before oversampling them.

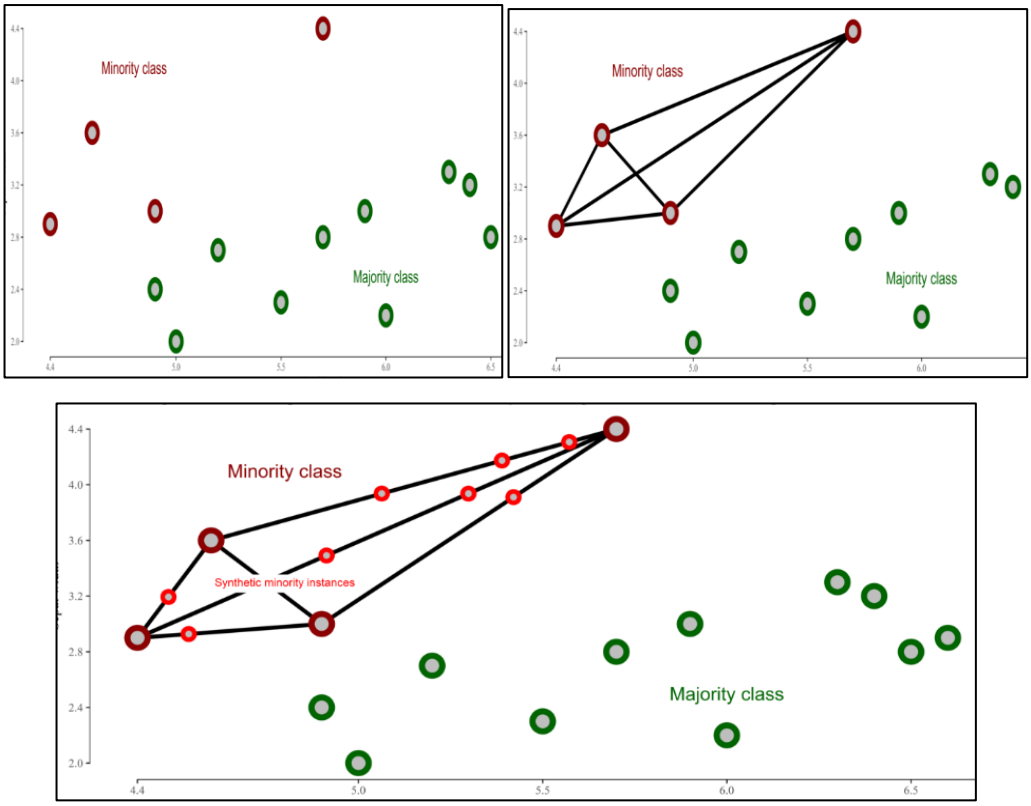


Figure 2.1: SMOTE methodology in generating synthetic data points (Source: http://rikunert.com/SMOTE_explained)

The minority examples which are categorized into 'Danger' are the borderline data of minority class and are oversampled using SMOTE approach. Synthetic data is generated by multiplying the difference between each instance of the 'Danger' category with its *s* nearest neighbors from minority class itself with a random number between 0 and 1. Synthetic examples are generated along the line between minority border line examples and their nearest neighbors of the same class, thus oversampling the borderline examples.

2.5.2 SMOTE- Borderline 2

In this technique, minority class examples are categorized into 'safe', 'noise' and 'danger' as categorized in previous approach. Synthetic data is generated by multiplying the difference between each instance of the 'Danger' category with its *s* nearest neighbors from minority class with a random number between 0 and 1. Likewise, Synthetic data is generated

by multiplying the difference between each instance of the 'Danger' category with its *s* nearest neighbors from majority class but with a random number between 0 and 0.5 which will be closer to the minority class. Therefore, oversampling the borderline minority examples.

2.5.3 Safe-level-SMOTE

Safe-level-SMOTE was proposed by (Bunkhumpornpat, Sinapiromsaran & Lursinsap, 2009) to overcome the drawback of SMOTE and Borderline-SMOTE in generating synthetic instances in an unsuitable region where overlapping and noise exists degrading the performance of the classifiers. Here, each synthetic instance is generated in safe region by considering safe-level ratio of the instances. Initially, safe-level (sl) which is the number of minority instances in k-nearest neighbor is calculated followed by safe-level ratio. If the safe level is closed to 0 then it is nearly a noise and safe if it is closer to k.

$$\text{Safe-level ratio} = \frac{Sln}{Slp}$$

where, Slp = the number of minority instances in k nearest neighbors for p in D
Sln = the number of minority instances in k nearest neighbors for n in D
p is the minority instances in D dataset
n is the selected nearest neighbors of p.

Five decisions are made on the values obtained for safe level ratio while oversampling minority examples and is given in the following table (table 1).

SAFE LEVEL RATIO	**P, N**	**INFERENCES**
Safe level ratio = infinity	p = 0	p and n are noises
Safe level ratio = infinity	p != 0	n is noise
Safe level ratio = 1	p = n	Synthetic instance will be generated along the line between *p* and *n* because p and n are both safe.

Safe level ratio > 1	p > n	synthetic instance is generated closer to *p* because *p* is safer than *n*.
Safe level ratio < 1	p < n	synthetic instance is generated closer to *n* because *n* is safer than *p*

Table 2.1 : Safe-level SMOTE decision table

The experiment had a better performance when compared to SMOTE and Border-line SMOTE indicating that synthetic instances generated in safe regions can improve prediction performance of classifiers dealing with class imbalance problem.

2.6 Synthetic Data Generation Oversampling Method: Hard to Learn Minority Samples

(Garcia, 2008) proposed ADASYN which is used to adaptively generate minority data samples which are hard to learn by classifiers than those which are easy to learn. It is used to reduce the learning bias introduced by the imbalanced dataset and to adaptively shift the decision boundary to focus on those difficult to learn sample. Initially, the count of minority and majority samples are calculated followed by degree of class imbalance using,

d=minority class count / majority class count

where, d € (1,0)

If the calculated degree of class imbalance is less than the threshold of maximum tolerated degree of imbalance, then the ADASYN approach is carried out which is as follows,

1. Calculate the number of synthetic data examples to be generated for the minority class using the following,

   G = (majority class count – minority class count) * β

   where, β € (0,1)

β is the desired balance level to be maintained after the generation of synthetic points. If β = 1, fully balanced dataset will be created.

2. For each instance in the minority examples, K nearest neighbors is calculated based on Euclidean distance in n dimensional space and the following ratio is calculated.

$$r_i = \frac{\Delta i}{K}$$

where, i = 1 to instance in majority example count,

Δ is the number of examples in K nearest neighbors that belongs to majority class examples.

3. Normalize r_i according to

$$\hat{r}_i = r_i / \sum_{i=1}^{minority\ data\ instances} ri$$

where, $\hat{r}_i$ is density distribution.

4. The number of synthetic data examples to be generated for each minority example is calculated using the following,

$$g_i = \hat{r}_i * G$$

5. Synthetic data examples are generated by multiplying the difference between each randomly chosen minority data point with its *k* nearest neighbors of the same class with a random number between 0 and 1. The original data is updated with the addition of oversampled data.

The experiment outperformed SMOTE in terms of G-mean and accuracy.

2.7 SMOTE + SVM Classifiers

2.7.1 SVM On Imbalanced Data

SVM when applied on highly imbalanced datasets, its performance deteriorates significantly. While training an imbalanced dataset using SVM, class-boundary learned by SVM will be severely skewed resulting in high false negative rate. However, SVM can

perform well with moderately imbalanced data sets and it is unaffected by non-noisy negative instances far away from the boundary regardless of its count as it only considers the instances which are close to boundary line. (Akbani, Kwek & Japkowicz, 2004)

Causes of performance loss with Imbalanced dataset are:

1. Positive Points Lie Further from the Ideal Boundary: In the case of imbalanced training data ratio wherein the negative instances outnumber positive instances, the minority data points might have situated farther away from the ideal boundary line as shown in the figure 2.2. Since, SVM considers the instances that is too close to the boundary, minority instances will be mis-classified as majority class instances.

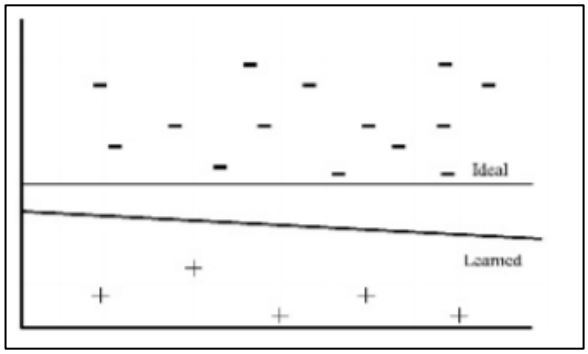


Figure 2.2: Positive instances lie further away from the ideal boundary (horizontal line) than the negative instances. As a result, SVM learns a boundary (slanted line) that is too close to the positive support vectors (Akbani, Kwek & Japkowicz, 2004)

2. Weakness of Soft-margins: SVM is more focused in minimizing the mis-classification error by maximizing the margin. During this course of time, the penalty constant, C the trade-off between the empirical error and the margin which minimizes the error should be tuned properly. It is advisable to set the values of C as high with respect to minority class instances.
3. Imbalanced Support vector ratio: As the imbalanced ratio increases, ratio of support vectors will become more imbalanced. The neighborhood of a test instance close to the boundary will be more dominated by (majority class) negative support vectors and hence the decision function is more likely to mis-classify boundary point as majority class. To avoid this problem, increasing the weight of the minority class instances is advisable.

2.7.2 Synthetic Informative Minority Over-Sampling (SIMO) Algorithm Leveraging Support Vector Machine (SVM)

A novel method 'Synthetic informative minority over-sampling (SIMO) algorithm leveraging support vector machine' is proposed by (Piri, Delen & Liu, 2018). SIMO's prime focus is on creating synthetic data points of informative minority data points alone. This can be achieved with the help of SVM, as data points that are close to the boundary of classes can be chosen as informative minority instances. The behavior of SVM on imbalanced datasets and use of these behavior in detecting informative instances during the application of SIMO is explained below.

The first and foremost step in performing SIMO is partitioning the dataset into train and validation dataset. To avoid bias, partition is carried out in such a way that constant imbalanced ratio is maintained in both train and validation dataset. Imbalanced gap which is the difference in count of minority and majority data instances is calculated for training dataset.

As briefed earlier, SVM misclassifies minority instances as majority instances when applied on highly imbalanced datasets. This might be because the minority data points lie further away from ideal boundary line due to which SVM tends to learn a boundary that is too close to the minority support vectors (Akbani, Kwek & Japkowicz, 2004). Thus, train data is trained using SVM and G-mean is computed. G-mean is the metrics which is usually calculated in the case of imbalanced data along accuracy. Higher the G-mean, better is the performance of the model.

Furthermore, Euclidean distance is calculated between the data point and the SVM decision boundary. As our major focus is informative minority data points and as mentioned earlier, data points close to the boundary of classes are important and informative and hence those data points with least Euclidean distance from decision boundary is identified as informative minority data instances. Using delta parameter, range of the data points with least Euclidean distance is selected as informative minority data points to oversample. Delta value is between 0 to 1 and for example if delta of value 0.2 is chosen, then top 20 percent of minority data points with least Euclidean distance value will be chosen for oversampling.

Synthetic data points are generated on the application of SMOTE oversampling technique on the informative minority data instances.

SIMO Algorithm

Given D, delta and p

Partition D into training and validation dataset.

1. Calculate Imbalanced gap, Imbalanced_gap = Majority_count – Minority_count
2. Develop initial SVM model on train dataset.
3. Compute Gmean ,

$$G - mean = \sqrt{(TPR * TNR)}$$

4. Initialize, SVM = initial_SVM,
   G_Mean=G_Mean_initial,
   generated_data_count =0
   While generated_data_count < Imbalanced_gap,
5. Calculate Euclidean distance of minority data points from Decision boundary.

$$Euc\_D(x^{k+}) = \frac{|\sum_{t=1}^{m} w_t x_t^{k+} + b|}{\sqrt{\sum_{t=1}^{m} w_t^2}}$$

6. Select top delta percent of minority data points close to boundary line based on calculated Euclidean distance. This will be regarded as informative minority data points. (delta = 0 to 1)
7. Oversample the chosen percent of minority data points using SMOTE approach.
8. Calculate the number of synthetic generated data points generated.
9. Update the initial training dataset with the resampled data.
10. Apply SVM and compute G_mean.
11. Update G_mean_initial with computed G_mean
12. End
13. Find the maximum G_mean and its index.
14. Select the oversampled training dataset with maximum G_mean.
15. Train the model of interest (Logistic regression and SVM linear) on final over-sampled training dataset
16. Evaluate the model on test dataset by computing G_mean and AUC.

Notations:

D – Imbalanced Data set

delta – Top delta percent of minority data points that are close to decision boundary

p – Oversampling degree for minority informative data points at each iteration

At this stage, the original imbalanced train data is updated with oversampled data. The number of synthetically generated data points and their indices will be recorded at each iteration. SVM will be applied on the updated training dataset. The decision boundary of this new SVM will be shifted toward the majority class data space closer to the ideal decision boundary as position of the SVM decision boundary only depends on the support vectors. The reason is that by generating synthetic minority examples, the imbalance ratio of the training dataset will be reduced along with an alleviated imbalance ratio of the support vectors. Performance of the SVM will be evaluated by computing the G mean and the Euclidean distance of the minority data points of an updated train dataset from the new SVM decision boundary is calculated, informative ones will be selected, and new synthetic minority data points will be generated. It is repeated until the number of synthetically generated examples reaches the imbalanced gap.

The performance of the model depends on the structure and complexity of the dataset. In each iteration, dataset will be updated with new synthetic generated minority examples. Though SVM improves on updated data set in each iteration, it isn't guaranteed the same for all kinds of datasets. Hence the performance parameters are recorded for each iteration and the iteration with higher G mean value is identified as the best performing model and the training dataset associated with that iteration will be selected as final oversampled training dataset. This novel approach performed well regardless of learner algorithms used and generated comparably less synthetic data instance than other oversampling methods as the focus was on oversampling informative instance alone and hence reducing the computational cost.

2.7.3 Biased Support Vector Machine (SVM) And Weighed-SMOTE

Similarly, (Hartono, Sitompul, Tulus & Nababan, 2018) proposed Biased support vector machine and weighed-SMOTE in handling class imbalance problem. It is the

combination of Biased SVM and weighed-SMOTE techniques. (Gonzalez-Abril, Angulo, Nuñez & Leal, 2017) designed a preprocessing technique of modifying the bias of a standard SVM to improvise its performance on imbalanced dataset by fixing minimum value for recall, to maximize specificity on the training set and named the outcome as BSVM (Biased SVM). It is designed for scenarios where it is non-critical to increase true positive rate in trade-off with increase in false positive rate. BSVM achieved better performance in sensitivity and reduction in accuracy and hence (Prusty, Jayanthi & Velusamy, 2017) used weighed-SMOTE along to overcome the drawback.

The working of weighed-SMOTE is as shown in the figure 2.3. The Euclidean distance of each minority sample is collected with respect to the other minority data samples and then they are normalized using min-max normalization which is carried out to fit in the distance values in the range of 0 and 1. Remodeled Normalized Euclidean distance(RNED) represents that lesser Euclidean distance more the share it gets to generate synthetic samples out of total synthetic samples needs to be generated and hence RNED matrix can be given as the difference between normalized ED value of each minority data and sum of all the normalized ED values. Furthermore, weight matrix is calculated by dividing each minority data share fraction from the total sum of the shares in the RNED matrix, based on which Smote generation matrix is obtained.

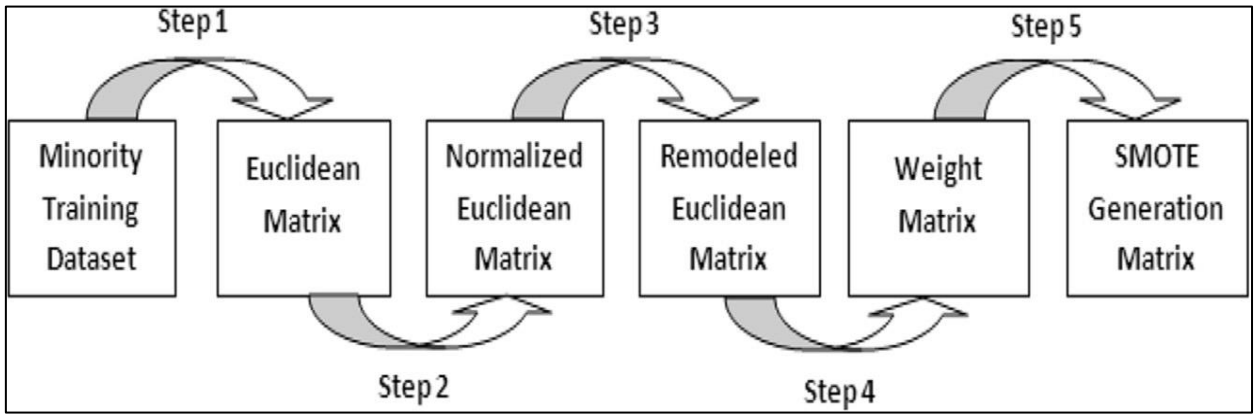


Figure 2.3: Block diagram of weighed-SMOTE (Hartono, Sitompul, Tulus & Nababan, 2018

In Biased support vector machine and weighed-SMOTE method, BSVM will classify classes into minority and majority support vectors(SV) sets and non-support vector(NSV) sets. Noises are removed from both minority and majority SV sets. NSV of Majority instances and SV of minority instances are processed using weighed-SMOTE approach. The NSV and SV of minority classes are combined to obtain new minority sample sets similarly the majority sample sets are obtained. The outcome of this experiment yielded satisfactory results in handling class imbalance with minority class in a high priority.

2.8 Cluster-Based Oversampling Methods

2.8.1 Cluster-SMOTE

An addition to the existing methods are cluster-based methods where the dataset is reduced to clusters of similar class examples and those clustered instances are oversampled or under sampled. These approaches are focused on oversampling hard-to-learn instances (important information for the classifiers) which are usually available near decision boundary or belong to small concepts in the datasets which is referred as within-class imbalance. Cluster-SMOTE was proposed by (Cieslak, Chawla & Striegel, 2006), where k-means clustering is applied on the set of minority instances in the dataset and then SMOTE is applied on each cluster to oversample generating synthetic examples, and these are updated into the original dataset. These are advantageous in deriving the class regions and their borders for small group of minority examples which is as shown in the figure 2.4.

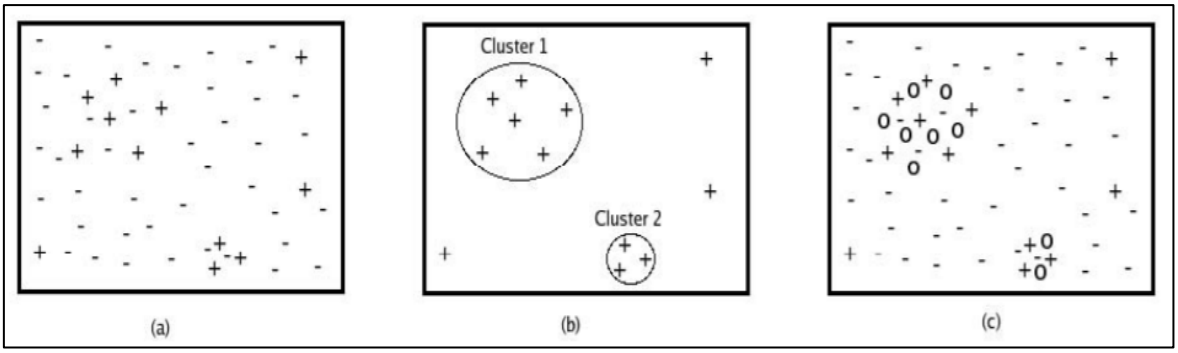


Figure 2.4: (a) features a sparse majority class, a minority class region, and many minority outliers. (b) Cluster-SMOTE detects two clusters of minority points and uses this information to generate new synthetic examples, as shown in (c) (Cieslak, Chawla & Striegel, 2006)

2.8.2 Adaptive semi-unsupervised weighted oversampling (A-SUWO)

A-SUWO a cluster-based oversampling method is proposed by (Nekooeimehr & Lai-Yuen, 2018). Minority instances are clustered using semi-supervised hierarchal clustering approach, the size of the sub cluster is determined based on the complexity of the sub-clusters in being mis-classified. It can be determined by a measurement parameter based on standardized average error rate which is obtained on cross validation. Synthetic examples are generated on assigning weights to the minority instances based on the average Euclidean distance to their NN-nearest majority neighbor. The advantage of this method is that it avoids generating overlapping synthetic data examples.

2.9 Oversampling + Under sampling approach

(Cateni, Colla & Vannucci, 2018) proposed a new sampling method named Similarity-based Under Sampling and Normal Distribution-based Oversampling (SUNDO) to balance the data without losing much of the significant data or adding up too much of unwanted synthetic patterns and it outperformed SMOTE approach. The method is the combination of both undersampling and oversampling techniques in achieving balanced data. Parameters k_0 and k_1, which represents minority and majority classes respectively are to be set along with the number of N samples should be removed. N is calculated as follows,

$$N = \text{round}\ (k_1 * n_0) - (k_0 * n_1)$$

3.0 Ensemble-based learning

Ensemble-based learning algorithms are playing predominant role in machine learning problems, especially while addressing class imbalance problem in many applications as it can improve the classification performance of any weak classifier. Ensemble learning is method of inclusion and combination of several classifiers to obtain new, better performing classifier. Ensemble based learning is the combination of ensemble based leaning and any of the imbalance learning approaches.

(Hao, Wang & Bryant, 2014) conducted an experiment with SMOTE coupled with GLMBoost to perform the classification of imbalanced datasets from PubChem BioAssay. The proposed experiment outperformed SMOTE in terms of performance metrices

Sensitivity and G-mean. Similarly, an empirical analysis to alleviate the class imbalance problem in heartbeat classification was carried out by (Rajesh & Dhuli, 2018), using data level sampling methods namely ROU, SMOTE+ RU and DBB and AdaBoost classifier and obtained significant performance measures. Henceforth, it can be concluded that, the application of data-level imbalance learning approach yielded better results with imbalanced dataset when combined with the learning algorithms irrespective of its complexity.

3.1 Summary of the Literature Review

The key focus of any sampling technique is to create a balance in the class distribution. Since, the dataset is small and highly imbalanced, the research is focused on the application of data-level imbalanced learning algorithms alone. Oversampling methods are advantageous over Under sampling as there will be no potential loss of information. Amongst the oversampling approaches, synthetic data generation is largely used, and its different approaches are reviewed in this section.

The advantage of SMOTE is, it broadens the decision region being less specific and generates artificial data points unlike Random Oversampling method, replicating the minority samples. Borderline-SMOTE majorly focus on those minority samples available near the borderline for oversampling. Since there is a suspicion of unsafe regions which would contain noisy data near the borderline, to ignore and oversample safe instances alone, Safe-level SMOTE was proposed. ADASYN is designed to detect and adaptively oversample those instances which are hard to learn by classifiers whereas SIMO using SVM oversamples informative data instance alone considering G-mean as the deciding performance metric which will be usually misclassified by the standard classifiers. Biased SVM with SMOTE is also designed for the same with an added tuning to the classifier SVM. Cluster oversampling focus on oversampling hard to learn instances which are found within-class imbalance whereas ensemble-based learners treats imbalance data by using ensemble learners along sampling approaches.

Although the methods are advantageous and yielded good performances there exist a shortcoming as well. Random oversampling might oversample noise minority data points

and is prone to problem of overfitting. The synthetic examples might be generated in overlapping and noise regions by Borderline-SMOTE. Problem of over generalization of synthetic data points that might lead to overlapping between classes can persist on the application of SMOTE. There can be an interpolation of a new sample between noisy data and one of its nearest neighbors in modified-SMOTE and ADASYN approaches. The generated minority instances in ADASYN will be slightly higher than majority instances which is in contrary with SIMO approach. This can have an influence on learner, training with high or insufficient samples. Also, the behavior of SVM on imbalanced dataset, role of SVM in improving the performance of imbalanced learning algorithms was discussed in this section.

In a nutshell, the extension and the adaption of SMOTE has produced improved results in terms of the model's performance in treating minority class examples.

3.2 Gap in the research

Non-representative small dataset can hinder the performance of the classifiers as training size must be quite large and large test sample is essential to accurately evaluate classifier with low error rate. According to (Raudys & Jain, 1991), small sample size and small disjuncts are closely related topics. The lack of data, small disjuncts and noisy data are claimed to be interrelated challenges faced by researchers in imbalanced classification (Fernandez, Garcia, Herrera & Chawla, 2018).

'Synthetic informative minority over-sampling (SIMO) algorithm leveraging support vector machine' technique is implemented in this experiment. The advantages of this method over other methods are low computational training cost as the amount of synthetic data generated will be less, avoidance of overfitting, focus on informative minority instances alone and the use of efficient learner SVM (Piri, Delen & Liu, 2018). However, the algorithm's performance is mainly dependent on distribution complexity and size of the datasets. Originally, SIMO has dealt with datasets of records ranging from 300 to 1500, yielding better performances when compared to other approaches. But the experiment is still not implemented on big data and very small dataset.

The values which should be considered for the parameters delta and p is specified in the paper with respect to the severity of class imbalance present in the data. It is advised to consider the values between 30 to 40 percent for delta and between 25 to 50 percent for p in high class imbalance scenarios. But these values being same for different size of the data is still in question. Performance of the classifiers can be influenced by size of the data as well and hence it is necessary to study, if performance of the classifiers on the implementation of this approach varies when applied on small datasets.

To study of the above-mentioned gaps following research question is proposed,

"*Can performance of the classifiers on small datasets, significantly improve on the application of 'SIMO leveraging SVM' over the application of baseline imbalanced learning algorithms?*"

***SIMO: Synthetic Informative Minority Over-Sampling*

***SVM: Support Vector Machine*

***Baseline imbalanced learning algorithms: SMOTE, SMOTE-Borderline1, SMOTE- Borderline2 and ADASYN.*

***Performance metrices: Accuracy, G-mean and AUC*

CHAPTER 3
EXPERIMENT DESIGN AND METHODOLOGY

This chapter will give an account of the plan and methodology used to answer the research question by implementing the process involved in CRISP-DM reference model, an overview of data mining project lifecycle. The research starts from business understanding followed by data understanding, data preparation, modeling and evaluation covering five phases of the reference model (figure 3.1). Scikit-learn machine learning package available in python programming language is used to obtain results in support of the decision to reject or retain the hypothesis.

The objective of this research is to evaluate the performance of 'Synthetic Informative Minority Over-Sampling leveraging SVM' on the datasets of records less than 150 with imbalance ratio ranging from 20 percent to 40 percent, over other imbalanced learning algorithms; SMOTE, SMOTE-Borderline and ADASYN. The balanced dataset is trained using base classifiers SVM-Linear and Logistic regression and its performance metrices are obtained for comparison. The methodology and design used to achieve the above, is explained in each section of this chapter with respect to the CRISP-DM framework in detail.

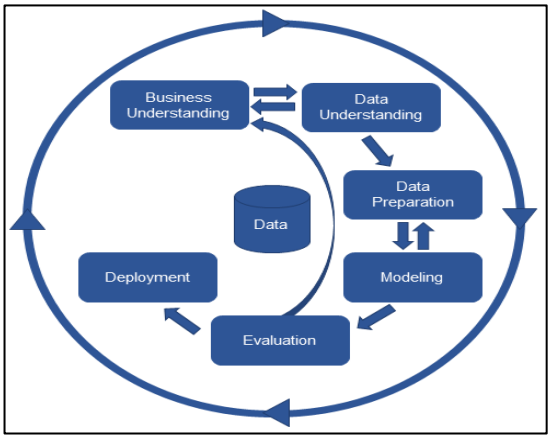


Figure 3.1: CRISP-DM framework

3.1 Business Understanding

The research is mainly focused on the performance of Synthetic informative minority over-sampling (SIMO) algorithm leveraging support vector machine on very small imbalanced dataset of records less than 150, since the original paper has dealt with datasets of records more than 300. This novel algorithm have had performed comparatively better than others in the original paper, thus the aim of this paper is to measure its performance on very small datasets. Imbalanced learning algorithms like SMOTE, SMOTE- Borderline and ADASYN are chosen as baseline to compare and evaluate the hypothesis. The hypothesis to address the research question is as follows,

H_0: "The G-mean and AUC of the models built on the oversampled datasets using imbalanced learning algorithm SIMO is equal to the G-mean and AUC obtained from the models on application of baseline algorithms SMOTE, SMOTE-Borderline and ADASYN, with p-value < 0.05."

** *SIMO: Synthetic Informative Minority Over-Sampling*

** *SMOTE - Synthetic Minority Over-Sampling Technique*

** *ADASYN - Adaptive Synthetic Sampling Approach*

** *AUC – Area Under Curve*

3.2 Data Understanding

Datasets

Four datasets are used in this research paper out of which three are taken from UCI repository and one is collected in a time span of 5 years (2004 - 2009) from an European hospital. These are highly imbalanced data with records less than 150 and is chosen to validate performance improvement of the models when built on oversampled dataset by using a novel imbalanced learning algorithm, S*ynthetic informative minority over-sampling (SIMO) leveraging support vector machine* over the models built by algorithms like SMOTE, SMOTE-Borderline and ADASYN which are readily available imbalanced learning packages in python. A brief description on the datasets is provided in the table below.

Dataset	**No. of records**	**No. of features**	**Imbalanced Ratio**	**Minority class**	**Majority class**
Biomarker	93	51	60:40	'Yes'	'NO'
Hepatitis	154	20	80:20	'DIE'	'LIVE'
Echocardiogram	131	13	70:30	'ALIVE'	'DEAD'
Immunotherapy	90	8	80:20	'NO'	'YES'

Table 3.1: Dataset description

Dataset description:

1. Biomarker: This dataset contains 93 records with 51 features with a class imbalance ratio of 60:40. The Target variable is 'Death' which represents the mortality risk in elderly patients as 'yes' or 'no' in presence of specific biomarkers which are the independent variables or predictors. The variable description is as shown in table 3.2 below.

Biomarker	Type	Description
age	N	<60, 60-65, 66-70, 71-75, 76-80, >80
sex	C	F- Female, M - Male
hyper	C	Diagnosis of Hypertension Low-grade (<160/90 mm Hg; medications are not used, or used irregularly), High-grade (>160/90 mm Hg; medications are used regularly)
DM	C	Diagnosis of Diabetes mellitus type 2 yes, Impaired glucose tolerance, No
HbA1c	N	Glycosilated Haemoglobin (%) - a marker of an average blood glucosein a three-month period
Fglu	N	Fasting glucose (mmol/L) - a marker of glucose metabolism
Chol	N	Total cholesterol
HDL	N	HDL-cholesterol
Statins	C	Statins use Yes, No
anticoag	C	Therapy with oral anticoagulant drug (warfarin), therapy with antiaggregant drug (aspirin), therapy with antiaggregant plant drug (ginkgo)
CVD	C	Cardiovascular disease. No, myocardial infarction/angina/history of revascularization, chronic myocardiopathy with atrial fibrillation, chronic myocardiopathy without atrial fibrillation, stroke/transient ischaemic cerebral event, carotid artery atherosclerosis confirmed by using image techniques, peripheral vascular disease
BMI	N	Body mass index (a measure of the body weight) <20, 20-25, 26-29, >=30
w/h	N	Weist to hip ratio - M <1.0, >=1.0; F <0.8, >=0.8
skinf	N	Triceps skinfold thickness Please, split the range of the values into tertiles, separately for M and F
COPD	C	Chronic Obstructive Pulmonary Disease – Yes, No
Aller d	C	Allergic disease (rhinitis/asthma) - Yes, No
Dr aller	C	Allergy to drugs - Yes, No
Analg	C	Long-term use of analgesics/nonsteroidal antiinflammatory drugs - Yes, No
Neo	C	No, malignant disease in a stable phase, skin malignancy
Derm	C	Chronic skin disorders Chronic dermatitis, dermatomycosis, No
OSP	C	Osteoporosis - an age-related disease affecting mostly women, characterized with increased bone fragility and susceptibility for fracture. Osteoporosis/osteopenia/no - of the radius bone; osteoporosis/osteopenia/no - of the vertebrae; ; osteoporosis/osteopenia/no - of the hip
Psy	C	Anxyety/depression, Parkinson's disease, cognitive impairment, no
MMS	N	Neuropsychologic test for screening on cognitive impairment "Mini Mental Score" <10 severe cognitive imapirment, 10 - 20 moderate, 21 - 24 mild, 25 - 30 normal cognition
CMV	N	Cytomegalovirus infection (specific IgG antibodies, IU/ml).
EBV	N	Epstein-Barr virus infection (specific IgG antibodies, IU/ml).
HPA	N	Helicobacter pylori infection (specific IgA antibodies, IU/ml).
LE	N	White blood cell (WBC) count (Leukocytes number × 109/L).
CRP	N	C-reactive protein (mg/L).
GAMA	N	Hiper-gamma-globulinemia (g/L) - a marker of chronic inflammation
MO	N	Monocytes % in WBC differential.
NEU	N	Neutrophils % in WBC differential.
LY	N	Lymphocytes % in WBC differential
E	N	Red Blood Cell (RBC) count (Erythrocytes number × 10 2/L)
HB	N	Hemoglobin (g/L).
HTC	N	Hematocrite (Erythrocyte volume blood fraction)
MCV	N	RBC Mean Cell Volume (fL).
FE	N	Serum iron (g/L).
ALB	N	Serum albumin (g/L).
Clear	N	Creatinine clearance - an indicator of chronic renal impairment (ml/s/1.73m)
HOMCIS	N	Homocystein (µmol/L)- sulphuric amino-acid
VITB12	N	Vitamin B12 (pmol/L)
FOLNA	N	Folic acid (mM/L)
INS	N	Serum fasting insulin (IU/ml)
CORTIS	N	Serum cortisol in the morning (nmol/L)

Table 3.2: Biomarker dataset, C – Categorical variables, N – Numerical variables

2. Hepatitis: The dataset contains 154 records with 20 features with a class imbalance ratio of 8:2. The Target variable is ‘Class’ which represents whether a patient with hepatitis is dead or alive. The variable description is as shown in table 3.3.

Attribute information:

VARIABLES	TYPE	DESCRIPTION
Class	C	DIE, LIVE
AGE	N	10, 20, 30, 40, 50, 60, 70, 80
SEX	C	male, female
STEROID	C	no, yes
ANTIVIRALS	C	no, yes
FATIGUE	C	no, yes
MALAISE	C	no, yes
ANOREXIA	C	no, yes
LIVER BIG	C	no, yes
LIVER FIRM	C	no, yes
SPLEEN PALPABLE	C	no, yes
SPIDERS	C	no, yes
ASCITES	C	no, yes
VARICES	C	no, yes
BILIRUBIN	N	0.39, 0.80, 1.20, 2.00, 3.00, 4.00
ALK PHOSPHATE	N	33, 80, 120, 160, 200, 250
SGOT	N	13, 100, 200, 300, 400, 500
ALBUMIN	N	2.1, 3.0, 3.8, 4.5, 5.0, 6.0
PROTIME	N	10, 20, 30, 40, 50, 60, 70, 80, 90
HISTOLOGY	C	no, yes

Table 3.3: Hepatitis dataset, C – Categorical variables, N – Numerical variables

3. Echocardiogram: This dataset contains 131 records with 13 attributes with a class imbalance ratio of 7:3. The target variable represents if the patient suffering from heart attack is dead (0) or is still alive (1). The description of independent variables is given in table 3.4.

VARIABLES	TYPE	DESCRIPTION
Survival	N	the number of months, patient survived if they are dead and has survived, if patient is still alive.
still-alive	C	0=dead,1=still alive
Age	N	age in years when heart attack occurred
pericardial-effusion	N	Pericardial effusion is fluid around the heart. 0=no fluid, 1=fluid
fractional-shortening	N	a measure of contracility around the heart lower numbers are increasingly abnormal
epss	N	E-point septal separation, another measure of contractility. Larger numbers are increasingly abnormal
lvdd	N	left ventricular end-diastolic dimension. This is a measure of the size of the heart at end-diastole. Large hearts tend to be sick hearts.
wall-motion-score	N	a measure of how the segments of the left ventricle are moving
wall-motion-index	N	equals wall-motion-score divided by number of segments seen. Usually 12-13 segments are seen in an echocardiogram. Use this variable INSTEAD of the wall motion score.
alive-at-1	C	Boolean-valued. Derived from the first two attributes. 0 means patient was either dead after 1 year or had been followed for less than 1 year. 1 means patient was alive at 1 year.

Table 3.4: Echocardiogram dataset, C – Categorical variables, N – Numerical variables

4. Immunotherapy: It is a dataset of 90 records and 8 attributes with class imbalance ratio of 8:2. The target variable represents the response status of the patients on Immunotherapy treatment (wart treatment). This helps medical professionals in proceeding with the treatment if there is a positive response to the treatment from patient which is denoted as 'Yes' or to stop the treatment if the response is negative, represented as 'No'. It saves time and money of both patients and the hospital.

VARIABLES	TYPE	DESCRIPTION
Sex	C	"Man" = 1, "Women" = 2
Age	N	Age in years
Time	N	Time elapsed before treatment (month)
Number_of_Warts	N	1 to 19
Type	C	Common, Plantar, Both
Area	N	Surface area of the warts (mm2) (6 – 900)
induration_diameter	N	5 - 70
Result_of_Treatment	C	Yes, No

Table 3.5: Immunotherapy, C – Categorical variables, N – Numerical variables

An overview on the dataset can be derived from the following,

1. Descriptive statistics: This is carried out to obtain the details on central tendency (mean), median, mode, Inter-Quartile range, range, standard deviation and the skewness of the variables.
2. Missing value analysis: gives an account of missing value count and its percentage in the dataset with respect to each variable.
3. Exploratory data analysis: includes histograms to understand the distribution of numeric variables and to find out if there is any presence of outliers, frequency plots to analyze the relationship between categorical and numerical variables and correlation matrix to understand the relationship between numerical variables.

Based on the insights obtained from the above analysis, data preparation is carried out which is explained in the next phase in detail. The above analysis and visualization is carried out using available functions and packages in the python scikit-learn machine learning library.

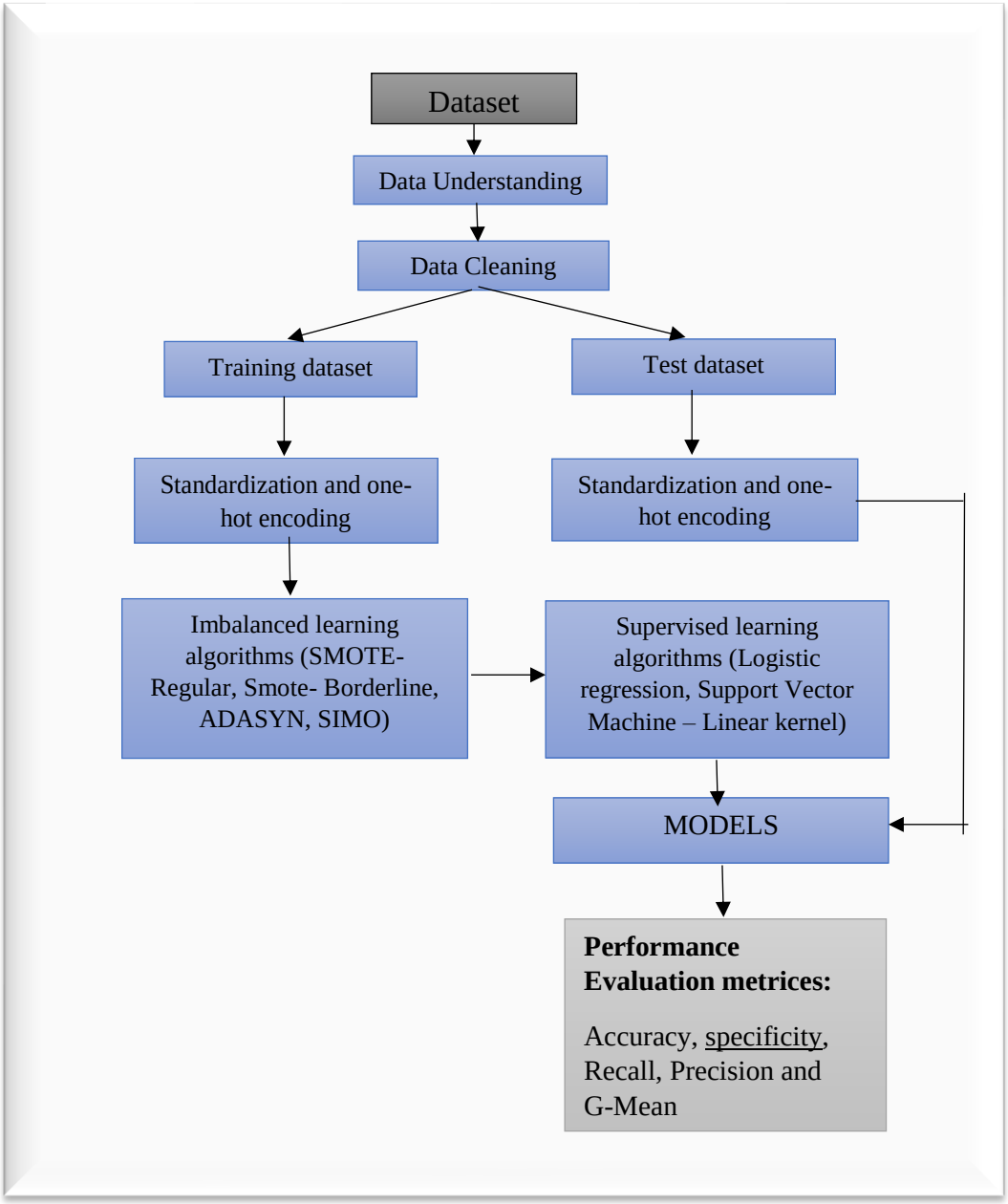


Figure 3.2: Design

3.3 Data Pre-Processing

In this phase, based on the understandings obtained from previous section, necessary data preparation methods are carried out which involves the following.

3.3.1 Data Imputation

Missing value analysis gives an overview on the missing value counts and its percentage with respect to each variable in the dataset. Data imputation is carried out on those whose values are missing. While imputation of numerical attributes, the histograms are also analyzed to detect if the data is prone to outliers as mean imputation will introduce a bias and is not advisable. On such cases median imputation will be carried out for continuous variables whereas mode imputation will be carried out for categorical variables.

3.3.2 Standardization: Z-Score

Numerical variables will have different impact on the predictive model in accordance with their ranges. Higher the range, higher the influence in prediction as predicters. Thus, the data should be scaled to fall under common range using Z-score standardization to improve the predictive accuracy. Standardization refers to shifting the distribution of each attribute to have a mean of zero and a standard deviation of one. It can be calculated as given below,

$$z - score = \frac{X - X'}{\sigma}$$

Where, X = sample

X' = mean of the sample

σ = standard deviation of the sample

The standard normal distribution of a dataset is as shown in the figure below (figure 3.3), which looks like a bell and thus called 'Bell Curve'. It has a symmetry about the center which is referred as 'mean' whereas standard deviation is the measure of quantifying the amount of dispersion of the data. Low standard deviation values usually will be closed to the mean whereas high standard deviation values denote how spread out the values are.

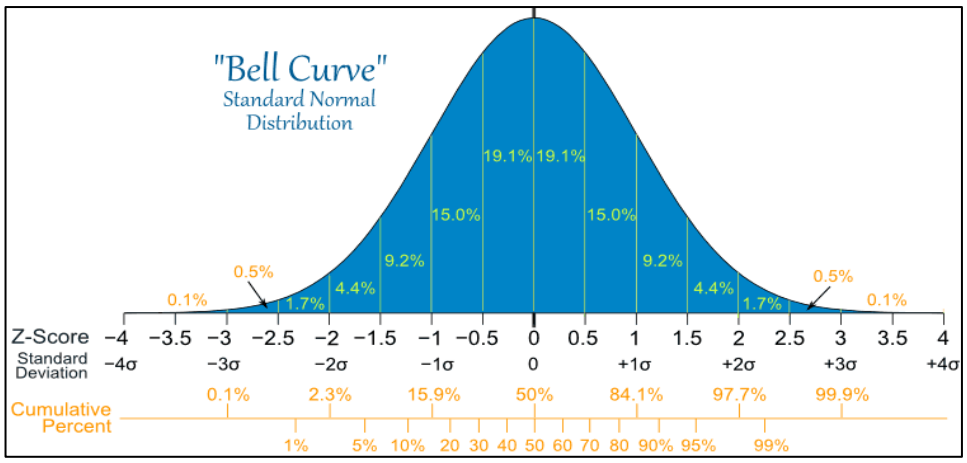


Figure 3.3: Bell Curve

3.3.3 One-Hot Encoding: Treating Categorical Variables

Datasets containing categorical variables should be treated before training models like Regression and Support vector machine. Otherwise, these variables will not make a meaningful contribution when trained with any learner based on standard distance metrics such as k-nearest neighbors as they get confused with the state of predictors. This can be eliminated by creating dummy variables, the process which is called one-hot encoding. This is carried out in the experiment by using function readily available in python's Pandas library named get_dummies().

3.3.4 Data Partition

Once the imputation of missing values, standardization of numerical variables and one-hot encoding of categorical variables are carried out, the data is partitioned into training and test dataset. Training data is used to train the model and the prediction is made on the test data. This avoids the problem of peeking in turn avoids overfitting. In this, experiment we have split the dataset into 75 percent of training data and 25 percent of test data with stratification of class variable to maintain the same imbalance ratio in test and train.

3.3.5 Imbalanced Learning Algorithms – Balanced Dataset

Since the chosen datasets are facing class imbalance problem; on modelling, the accuracy obtained will be inaccurate because of the bias introduced while training the model and the cost of misclassifying minority classes will be high. Thus, to overcome such problem the dataset should be balanced either by oversampling minority class or under sampling majority class in the dataset. In this experiment, oversampling is carried out using a novel technique named SIMO and its performance is compared with baseline imbalanced learning algorithms SMOTE, Borderline-SMOTE and ADASYN readily available in python scikit-learn library.

3.4 Modelling

Logistic regression and SVM are the base classifiers used to obtain predictive models in the experiment as the target variable is dichotomous (binary) in nature. These are commonly used classifiers in classification problems regardless of the records the dataset contains. Since the objective is to compare the performance of imbalanced learning algorithms used in balancing the dataset, these two basic supervised learning algorithms are chosen. Logistic regression tries to optimize loss function of training data by minimizing the least-square error and Support vector machine by regularized hinge loss.

Logistic regression is a statistical model used when the target variable is binary (0,1) and explains probability of an event, which is the linear combination of dependent and independent variables (figure 3.4). It is the task of estimating log odds of an event and is usually prone to overfitting on addition of more variables into the model. The logistic function is of the form,

$$h\theta(x) = \frac{1}{1 + e^{-z}}$$

where, z is logit

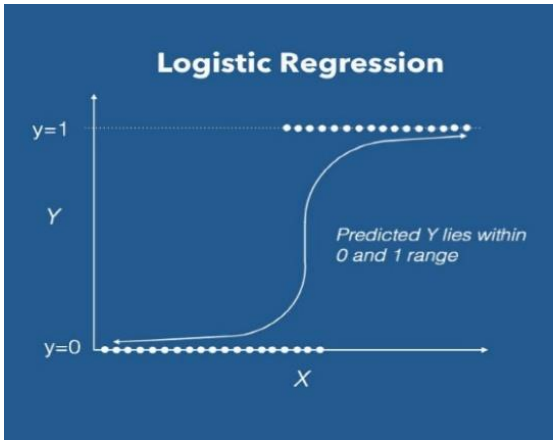


Figure 3.4: Logistic regression

Support Vector machine (SVM) is a statistical learning theory developed by the Russian scientist Vladimir Naumovich Vapnik and colleagues in 1962. It is generally used to tackle regression and classification problems. Support vector machines are computational algorithms that creates an optimal hyperplane or a set of hyperplanes in a high or infinite dimensional space that separates positive and negative instances with a maximum margin (figure 3.6). They are advantageous over other classifiers in terms of accuracy, robustness and efficacy in working on small data sets. Also, it shows greater ability in generalization and aims at minimizing mis-classification errors by maximizing the margin between separating hyperplane and the datasets. Its application ranges from image retrieval, handwriting recognition to text classification.

Linear SVM decision boundary is as shown in the figure 3.5. Classes in the dataset is represented as dots and stars. The data points which lie on the margins at either side of the decision boundary which are marked within circles are called support vectors. w is found normal to the decision boundary and b/|w| is the perpendicular distance of the decision boundary from the origin. The distance of a point represented as star which is misclassified as dot (as shown in the figure 3.5) from the decision boundary can be written as $-\varepsilon/|w|$. The decision boundary of the SVM can be formulated as $w^T x + b = 0$.

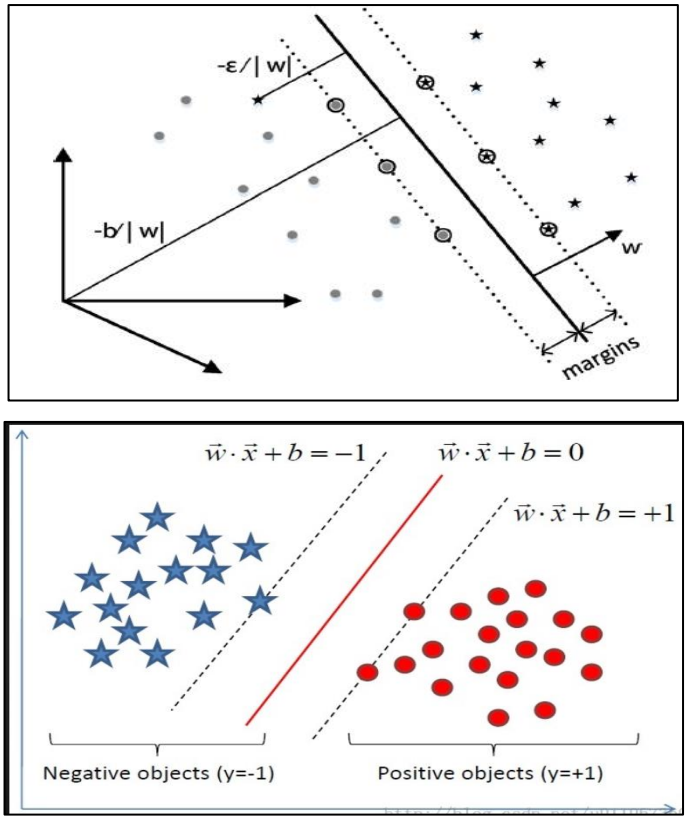


Figure 3.5: w is a weight vector, x is input vector and b is the bias.

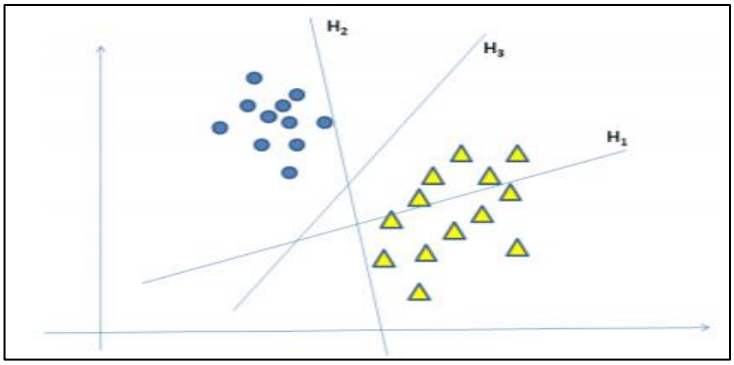


Figure 3.6: Linear Classification: H1, H2, H3 - Separation hyperplanes. H1 does not separate the two classes; H2 separates but with a very tinny margin between the classes and H3 separates the two classes with much better margin than H2

3.5 Evaluation

Performance evaluation of the model is conducted using the evaluation parameters; Accuracy, Precision, Recall and F- score. **Accuracy** given an account of correctly predicted observations amongst total observations. **Precision** defines how many of the positively classified instances are relevant. **Sensitivity/recall** explains how good a model is at detecting the positives. **F-score** is the harmonic mean of precision and recall. It is a measure of test accuracy. These parameters can be calculated using Confusion matrix.

Confusion matrix:

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

To plot ROC:

$$FPR = \frac{FP}{FP+TN} \quad , \quad TPR = \frac{TP}{TP+FN} \tag{9}$$

Metrices:

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

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

$$Recall = \frac{TP}{TP+FN}$$

$$F\ score = 2 * \frac{precision*recall}{precision+recall}$$

Where ,

- True positive (TP): actual and predicted target values, both are true.
- True negative (TN): actual and predicted target values, both are false.

- False negative (FN): actual value is true but predicted target value is false.
- False positive (FP): actual value is false but is predicted as true.
- TPR: True Positive rate, FPR: False Positive rate.

While evaluating the performance of the learner in imbalanced datasets, along with the generally used metric 'Accuracy', 'G-mean' should be measured as well. G-mean is the geometric mean of positive and negative accuracy which is TPR and TNR respectively. While training imbalanced dataset, as majority class examples outnumbers minority examples learner will tend to misclassify minority class examples as majority class examples. In such cases, the prediction accuracy will be higher which is quite misleading. Thus, along with Accuracy it is safe to consider G-mean whose values will be low if the model's performance is poor on either positive or negative examples.

$$G - mean = \sqrt{(TPR * TNR)}$$

Performance evaluation can also be carried out using Receiving Operator Characteristic chart. It is the plot of true positive rate (TPR) against the false positive rate (FPR) at various threshold settings which in turn is used to discard suboptimal models and select the optimal models which reduces the cost. Good model is expected to have higher TPR value and less FPR. ROC chart is as shown in the figure 3.7, where diagonal divides ROC space and points above diagonal signifies good classification results and its contradict by those which lie below the diagonal. It is a visual representation of trade-off between benefits and costs.

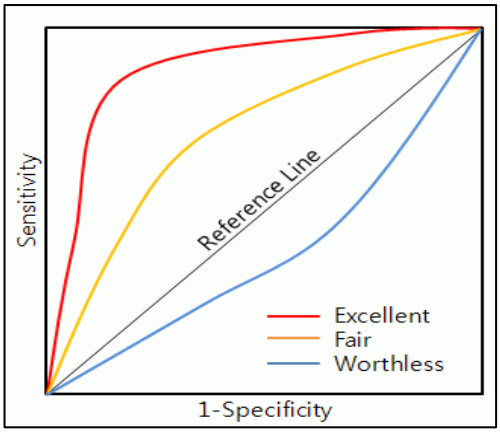


Figure 3.7: ROC chart

The G-mean and accuracy values obtained for logistic regression and SVM models on application of imbalanced learning algorithms SMOTE, SMOTE-borderline, ADASYN and SIMO is tabulated. The statistical significance of each experiment in support of rejecting or retaining the hypothesis is found using non-parametric statistical test `Wilcoxon Signed-Rank Test' with *p* value set to 0.05. It is used to compare the repeated measurements on single sample to asses if there is any difference. If the results obtained are statistically significant ($p < .05$) then null hypothesis will be rejected concluding that there is a performance difference on the application of SIMO over the application of other baseline imbalanced learning algorithms like SMOTE, SMOTE- Borderline and ADASYN on the datasets.

3.6 Strengths and Limitation

The section summarizes the strength and limitations of the design and methods used in the experiment. The design is proposed to evaluate the performance of novel imbalanced learning approach SIMO on small datasets of record less than 150 as approach is tested on dataset with the records in the range 300 to 1500. Since the chosen dataset is very small and faces class imbalance problem, classifier might not perform better as there is an inadequate information available for learning. As datasets chosen are underrepresented, obtaining

better predictive model for the same is quite challenging. Usually, data gathered from the domains like biomedical and clinical trial results often face the problem of having small sample size and class imbalance which is the motive of this experiment.

Dataset is partitioned into 75 percent of train set and 25 percent of test set. Stratified sampling is carried out as we are dealing with class imbalance problem. Usually while dealing with small datasets other data partition approaches like leave-one-out cross validation, k-fold cross validations are highly advisable, but here the data samples are quite small and training data is again split into 80 percent train and 20 percent validation set in method SIMO. This can pose scarcity of data and the classifier might not get sufficient data to learn and end up overfitting the model or performing worst. The entire design approach is executed five times in tradeoff with the cross-validation approaches to evaluate the reliability of the models.

Another limitation of this design is all the variables present in the datasets are chosen as predictors. No feature selection is carried out on the dataset to consider relevant and related predictors in the prediction of target. The reason is again the size of the data and the number of attributes it is dependent on. Losing potentially informative data isn't in the interest of this experiment and the chosen datasets have limited attributes and thus feature selection isn't termed mandatory.

Amongst the imbalance learning approaches discussed in literature earlier, SIMO is chosen because of the less computational training cost it will incur and avoidance of overfitting and overgeneralization which are the drawbacks of other approaches. Its major focus is on informative data instances and leveraging SVM in the approach is advantageous in obtaining those instances for oversampling. Also, the parameters included in the approach namely 'delta' and 'p' make it flexible to oversample the data to desired amount and to carefully select the percentage of informative class instances closer to the SVM decision boundary to oversample. Also, only those samples on which SVM had a better performance (evaluated by performance metric G-mean in each iteration till the imbalanced gap is met) is chosen.

GridSearchCV is used to choose the hyperparameters for Logistic regression and SVM to optimize their performance. However, use of SVM is limited to linear kernel alone and other kernels are not used in this experiment. Initially the idea is to keep the approach simple and study the behavior of base classifiers on the application of SIMO. This can also be a hindrance in studying the advantage of SIMO thoroughly. Performance metrices G-mean and ROC- AUC plot is obtained in addition with accuracy since there exist class imbalance problem. As design is executed five times, average of these metrices values are computed and used for comparison. Also, to support this, a non-parametric statistical test `Wilcoxon Signed-Rank Test' with *p* value set to 0.05 is carried out to find out if there is any difference in model's performance on the application of chosen imbalance learning algorithms.

CHAPTER 4

IMPLEMENTATION, RESULTS AND DISCUSSIONS

The research is carried out to evaluate the performance of novel imbalanced learning algorithm, Synthetic informative minority over-sampling (SIMO) leveraging support vector machine on very small datasets of records less than 150. The proposed design is repeated for five times and the average values of performance metrices are obtained to evaluate the reliability of the models when trained on different train-test sets. The performance comparison is carried out with other baseline imbalanced learning algorithms like SMOTE, SMOTE- Borderline and ADASYN. The results obtained on implementing the design explained in previous section as shown in (Figure 3.2) is covered in this chapter.

4.1 Data Understanding

The section includes Descriptive statistics and Missing value analysis report of each dataset used in the experiment. Also, the visual representation of each variable is provided for data distribution analysis, outlier detection and correlation analysis.

4.1.1 Biomarker

The descriptive statistics results for data ‘biomarker’ is as shown in the tables 4.1, 4.2, 4.3 &4.4. From table. 4.5, it can be observed that, variables are having a very less percent of missing values which can be imputed. Median Imputation is carried out on numerical variables to avoid the impact of outliers on the same whereas Mode imputation on categorical variables. Once the imputation is carried out, the distribution plots of each numerical variables are obtained to investigate if they are normally distributed or not. From figure. 4.1, it can be concluded that variables Age, chol, skinf, w_h, NEU, LY, HB, HTC, Clear, FT3 and FT4 are normally distributed with higher skewness with traces of outliers. To check if non-normally distributed variables can be fit into normal distribution on transforming, Standardization is carried out. No much improvement is found, and hence Non-parametric tests are carried out in future to derive correlation and other statistical results.

	ID	age	Fglu	HbA1c	Chol	HDL	BMI	wh	skinf	MMS	CMV
count	93	93	93	93	93	93	93	93	93	93	93
mean	47	67.7	6.52	5	6.18	1.44	29.04	0.951	33.36	25.12	6.22
std	26.99	7.96	2.1	4.72	1.38	0.37	4.2	0.066	7.377	3.48	3.6
min	1	47	4.6	2.89	3	0.87	20.24	0.76	16	14	0.06
25%	24	63	5.3	3.82	5.2	1.14	26.33	0.91	28	23	3.4
50%	47	68	5.7	4.14	6.2	1.4	29.38	0.95	33	25	5.7
75%	70	73	6.7	4.55	7.1	1.7	31.65	1.01	39	28	8.1
max	93	89	13.9	48.3	8.9	2.53	43.1	1.11	50	30	17.8

Table 4.1: Descriptive statistics of variables

	EBV	HPA	LE	MO	NEU	Ly	CRP	E	HB	HTC	MCV
count	93	93	93	93	93	93	93	93	93	93	93
mean	135.58	32.6	6.62	8.09	51.95	35.45	5.34	4.32	134.70	0.39	91.03
std	50.23	51.4	1.51	2.25	9.83	8.99	3.45	0.41	12.45	0.032	5.02
min	15.8	2.3	4.06	3.6	28	18.4	0.8	2.63	91	0.28	66.5
25%	121.2	6.2	5.61	6.5	45.4	27.8	3.8	4.08	127	0.37	88.6
50%	170	9.8	6.81	8	50.9	35.5	4.4	4.36	136	0.4	90.9
75%	170	29.5	7.65	9.3	59	42.2	5	4.57	141	0.41	93.7
max	170	200	9.93	15.7	73.3	57.7	24.5	5.37	167	0.47	106.1

Table 4.2: Descriptive statistics of variables

	FE	ALB	Clear	HOMCIS	RF	VITB12	FOLNA	INS	CORTIS	PRL
count	93	93	93	93	93	93	93	93	93	93
mean	14.68	46.1	1.69	12.35	27.8	284.33	20.71	23.21	374.59	124.57
std	5.14	3.13	0.45	3.80	82.86	158.78	8.52	17.08	122.38	120.38
min	5	33.1	0.72	5	9	97.8	6.5	5.9	180.3	14.57
25%	10.9	44.4	1.41	9.6	9	197	15.1	14.9	278.9	56.84
50%	14	46.5	1.63	11.8	9	247	19.5	18.9	362.6	105.83
75%	18	48.2	2.01	14.5	9	306	25	26.6	433	137.85
max	40	53	3.21	25.9	677	885.6	43.9	149.7	812.1	838.18

Table 4.3: Descriptive statistics of variables

	TSH	FT3	FT4	GAMA	ANA	IGE
count	93	93	93	93	93	93
mean	2.03	5.46	14	12.47	29.08	135.91
std	2.59	0.53	2.21	2.29	34.98	245.58
min	0.024	4.35	8.92	7.9	7	2
25%	0.96	5.17	12.3	10.9	14	16.3
50%	1.47	5.42	13.8	12.3	18.9	57
75%	2.19	5.72	15.9	13.7	32	143
max	22.7	7.96	18.9	21.3	300	1782

Table 4.4: Descriptive statistics of variables

Numerical	Missing	Percent
OSP	18	0.193548
anticoag	1	0.010753
Death	0	0
Psy	0	0
neo	0	0
derm	0	0
analg	0	0
draller	0	0
allerd	0	0
COPB	0	0
CVD	0	0
statins	0	0
DM	0	0
hypert	0	0
Sex	0	0

Categorical	Missing	Percent			
TSH	1	0.010753	Ly	0	0
Clear	1	0.010753	E	0	0
CRP	1	0.010753	ANA	0	0
IGE	0	0	FOLNA	0	0
MMS	0	0	GAMA	0	0
NEU	0	0	FT4	0	0
MO	0	0	FT3	0	0
LE	0	0	PRL	0	0
HPA	0	0	CORTIS	0	0
EBV	0	0	INS	0	0
CMV	0	0	VITB12	0	0
skinf	0	0	HB	0	0
wh	0	0	RF	0	0
BMI	0	0	HOMCIS	0	0
HDL	0	0	ALB	0	0
Chol	0	0	FE	0	0
HbA1c	0	0	MCV	0	0
Fglu	0	0	HTC	0	0
age	0	0	ID	0	0

Table 4.5: Missing value Analysis report

Spearman Correlation statistical test is carried out to analyze the relationship between variables and the target. A medium negative correlation exists between variables ‘skinf’ and ‘age’, ‘skinf’ and ‘MO’, ‘age’ and ‘clear’, ‘HOMCIS’ and ‘VITB12’, ‘HOMCIS’ and ‘FOLNA’. A strong negative correlation exists between ‘Clear’ and ‘HOMCIS’, ‘E’ and ‘MCV’, ‘NEU’ and ‘LY’. Target variable share weak correlation with the independent variables. (Correlation matrix results are provided in the appendix)

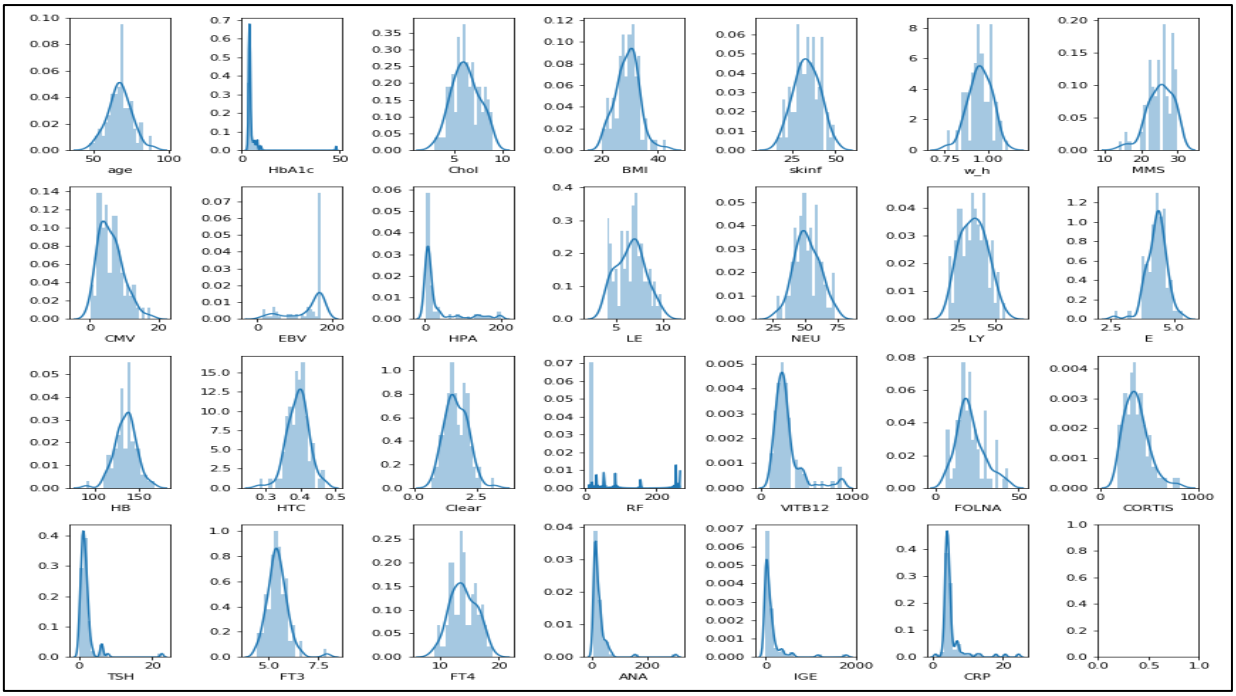


Figure 4.1: Histogram plots obtained before Normalization – Normality check

From fig. 4.3, it is found that, the categories of the Target variable 'Death' represented as 'TARGET' in the graph has a class imbalance where number of examples of class 0 exceeds the other. If there is a trace of biomarker 'aller_d' in an elderly patient, mortality risk is null which is not so representative of the behavior of the variable. Also, the relationship between target and variables follows class imbalance and hence biased information is observed.

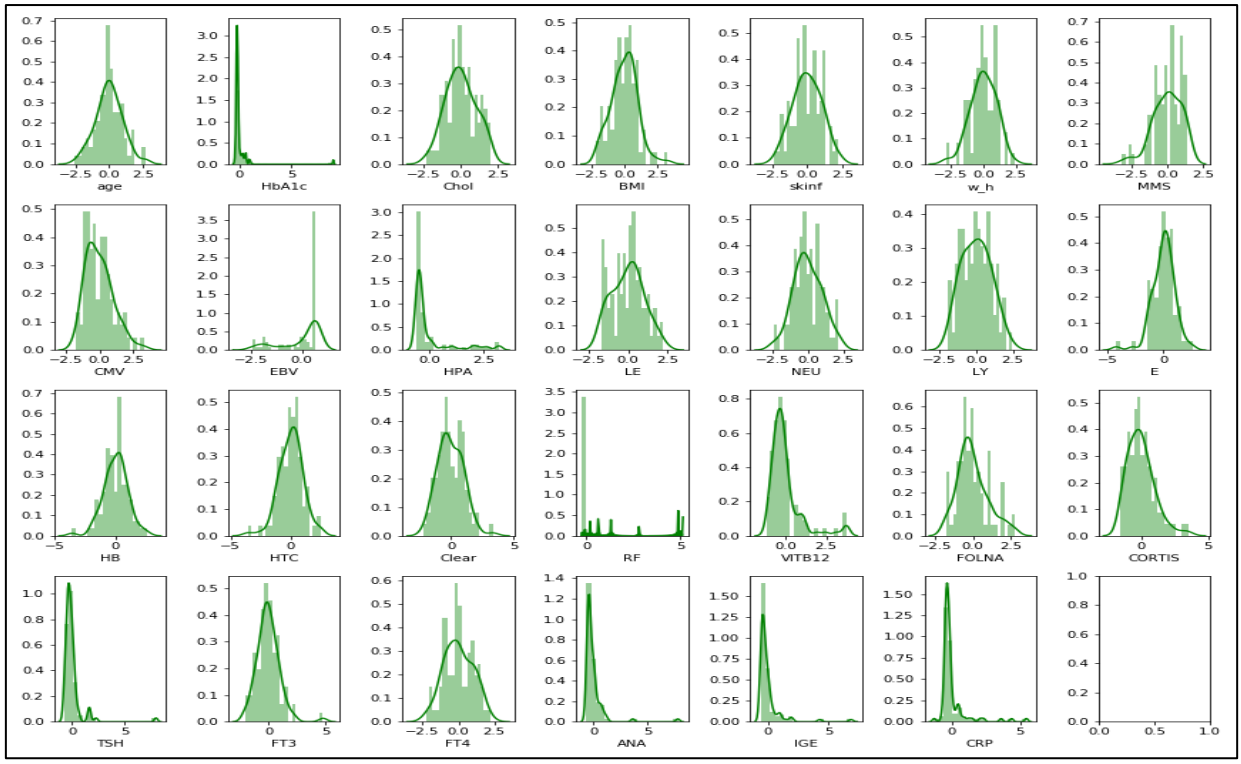


Figure 4.2: Histograms obtained after Normalization - Normality check

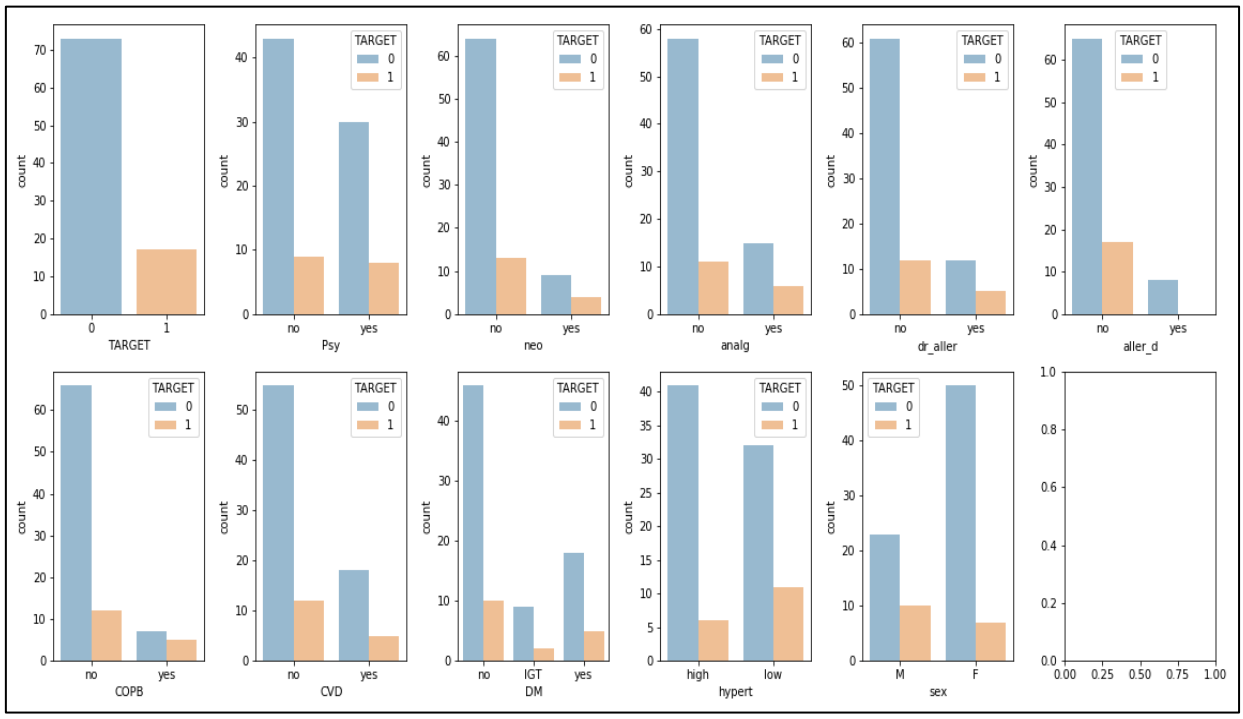


Figure 4.3: Frequency plot - Relation between Categorical and target variable

4.1.2 Hepatitis dataset

Descriptive Statistics is obtained as shown in the table 4.6. The count is same as the records in the dataset because the missing values are denoted by '?', as stated in data description retrieved from UCI repository which serves as the source of this dataset. Thus, empty class or any with '?' and 'NA' are replaced with value 'nan'. On missing value analysis (table 4.7), it is observed that variable PROTIME have more count of missing values followed by ALK_PHOSPHATE, ALBUMIN, LIVER_FIRM and LIVER_BIG which should be treated with necessary imputation methods. Median Imputation is carried out on numerical variables to avoid the impact of outliers whereas Mode imputation on categorical variables.

	Class	AGE	SEX	ANTIVIRALS	HISTOLOGY
count	154	154	154	154	154
mean	1.79	41.27	1.097	1.84	1.45
std	0.40	12.57	0.29	0.36	0.49
min	1	7	1	1	1
25%	2	32	1	2	1
50%	2	39	1	2	1
75%	2	50	1	2	2
max	2	78	2	2	2

Table 4.6: Descriptive Statistics

	Missing	Percent
PROTIME	66	0.42
ALK_PHOSPHATE	29	0.18
ALBUMIN	16	0.10
LIVER_FIRM	11	0.07
LIVER_BIG	10	0.064
BILIRUBIN	6	0.038
SPLEEN_PALPABLE	5	0.032
SPIDERS	5	0.032
ASCITES	5	0.032
VARICES	5	0.032
SGOT	4	0.025
ANOREXIA	1	0.0064
MALAISE	1	0.0064
FATIGUE	1	0.0064
STEROID	1	0.0064
SEX	0	0
AGE	0	0
HISTOLOGY	0	0
ANTIVIRALS	0	0
Class	0	0

Table 4.7: Missing value analysis report

Histogram of each continuous variable is obtained to analyze the distribution of the data (fig. 4.4). It is found that variable AGE and PROTIME are normally distributed, however kurtosis value of the variable PROTIME is high. The remaining continuous variables have skewness and should be normalized. On transforming numerical variables into its standardized form, following plot is obtained(figure) and no much improvement is found. As the variables follow non-normal distribution of data, non-parametric statistical tests are conducted in finding the correlation between independent variables, target and itself. From figure it is observed that there exists a class imbalance as the count of category '2' in target variable 'Class' outnumbers the other. The representation of categorical variables follows the same distribution fashion as variable 'Class' which isn't much informative thus there exists a need of oversampling the training data.

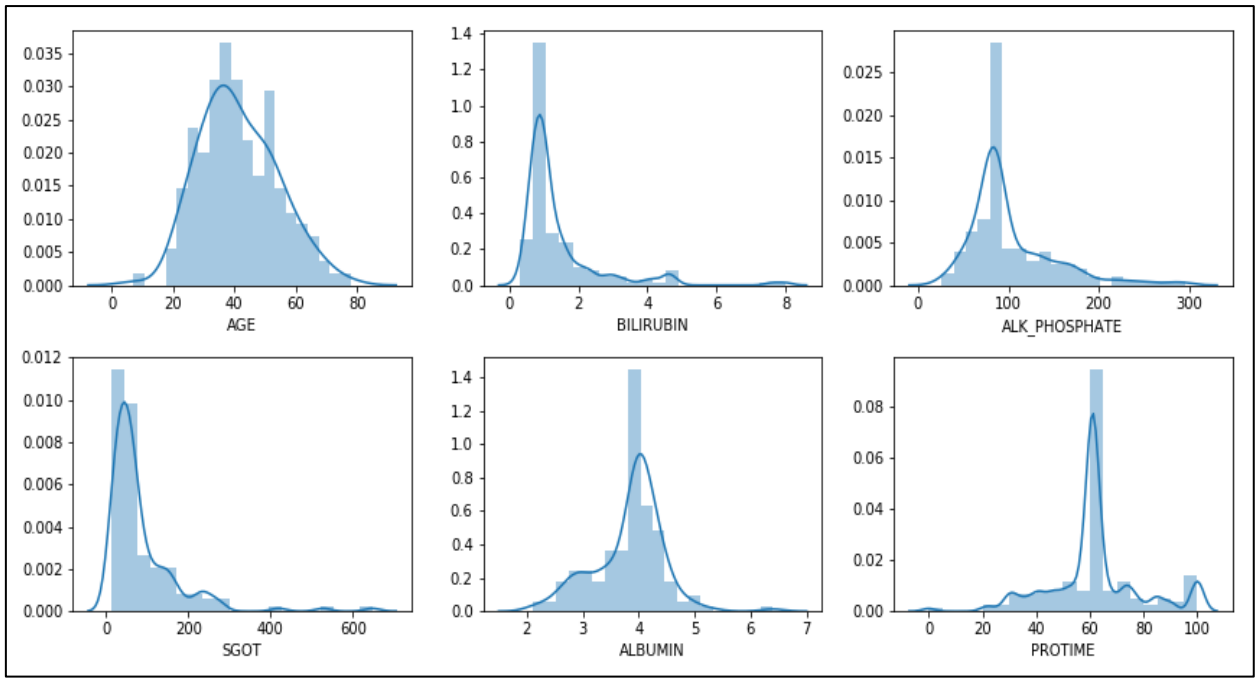


Figure 4.4: Histogram plots obtained before Normalization – Normality check

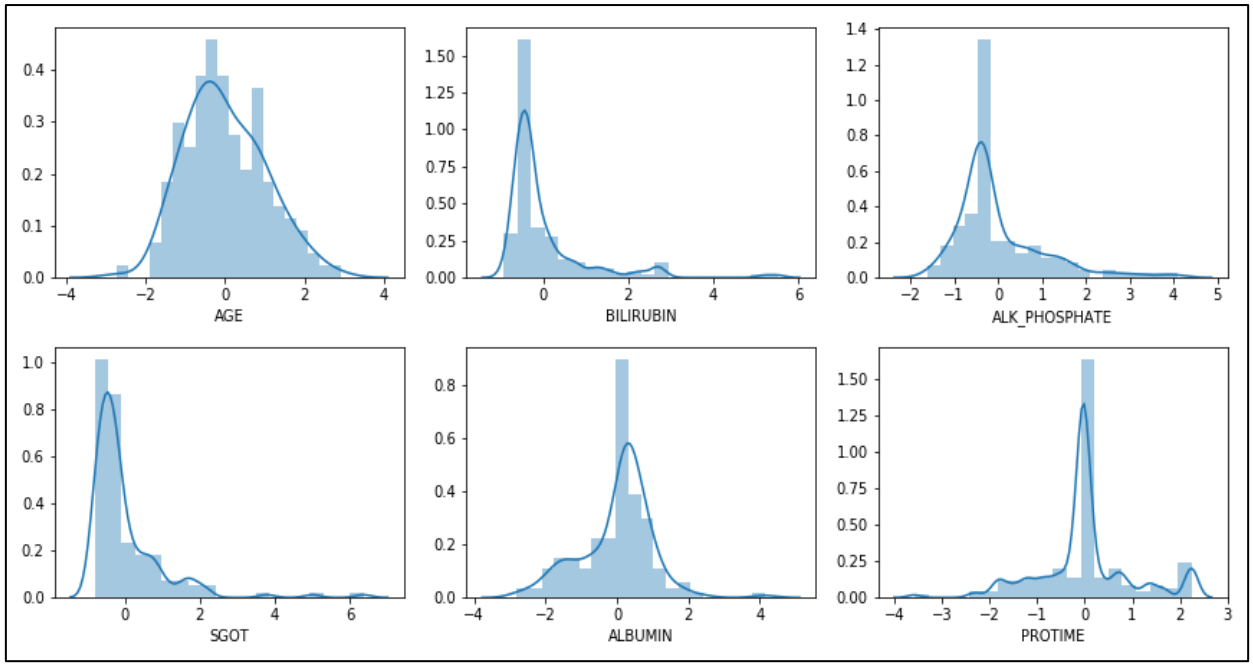


Figure 4.5: Histogram plots obtained after Normalization – Normality check

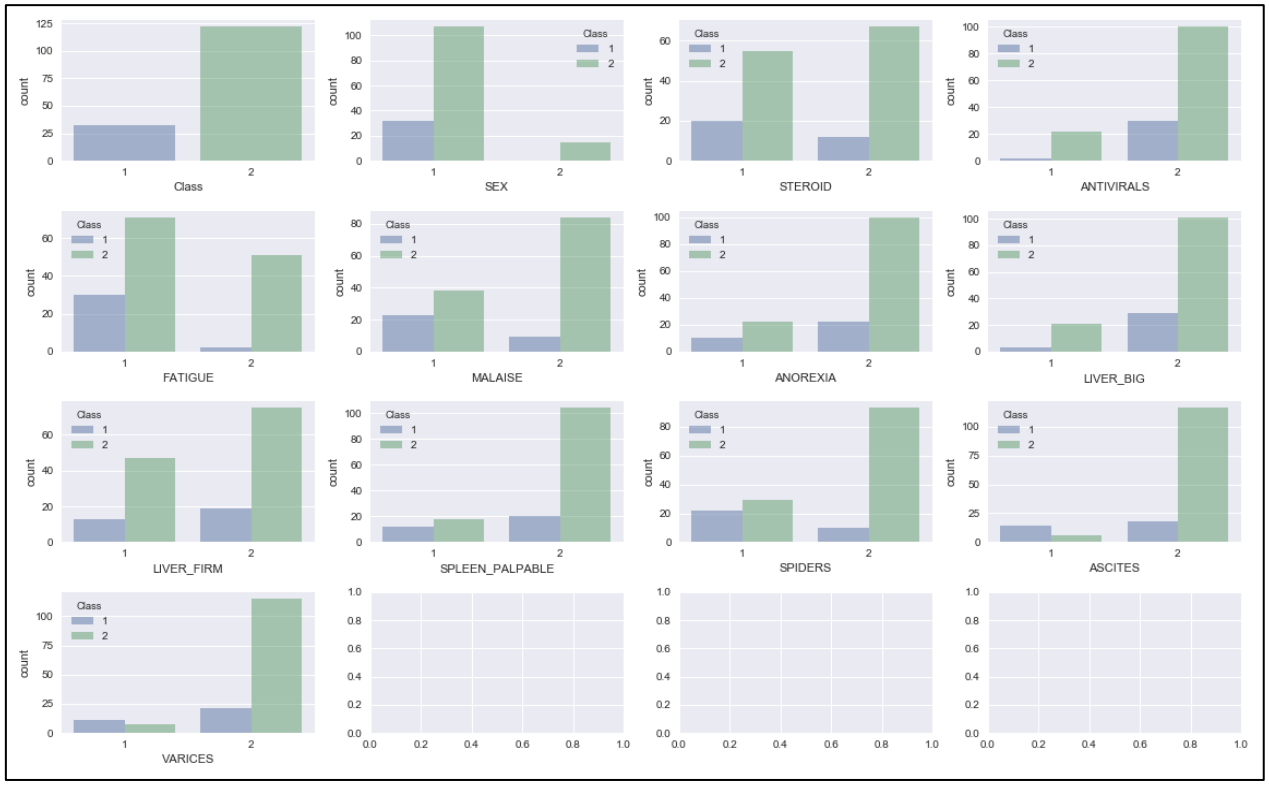


Figure 4.6: Frequency plot - Relation between Categorical and target variable

Spearman correlation, a non-parametric statistical test is carried out to find out if there is any correlation between the independent variables and the target which is represented through a heat map as shown in figure. Following observations are obtained,

1. A moderate positive correlation between the target ‘Class’ and independent variable ‘ALBUMIN’.
2. A negative moderate correlation between the target and ‘BILIRUBIN’.
3. A moderate correlation between independent variables ‘ALBUMIN’ and ‘PROTIME’ similarly between ‘BILIRUBIN’ with ‘SGOT’.
4. Variable ‘ALBUMIN’ have a moderate negative correlation with ‘BILIRUBIN’ and ‘ALK_PHOSPATE’.
5. Negative moderate correlation between ‘BILIRUBIN’ and ‘PROTIME’.

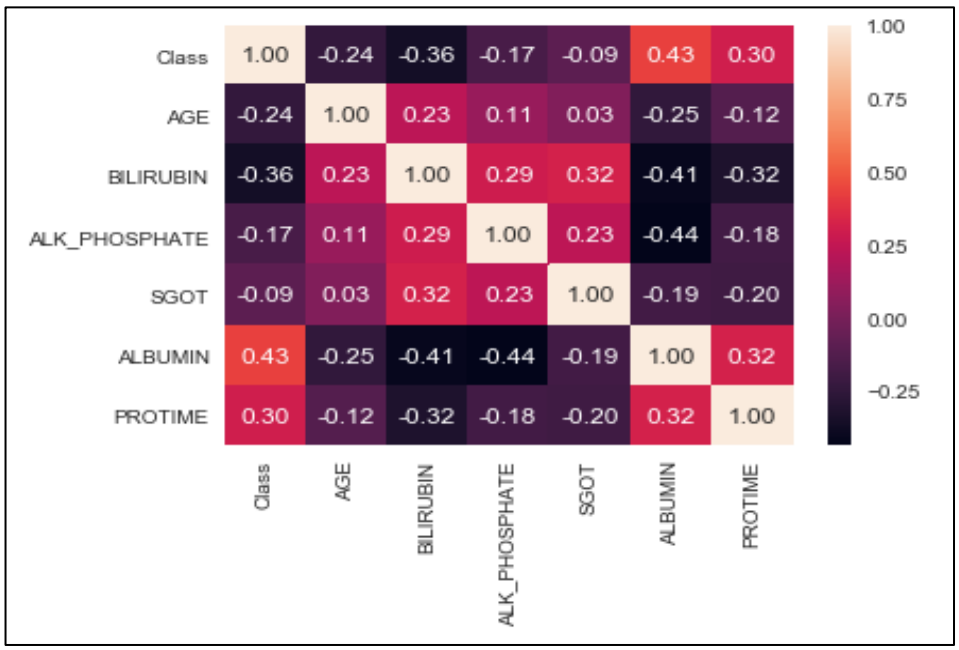


Figure 4.7: Correlation matrix represented in a heat map.

4.1.3 Echocardiogram

From the Descriptive statistics, central tendency, range, standard deviation is obtained for numerical variables as shown in the table 4.8. On finding the unique values

present in each variable unexpected value ‘2’ and ‘77’ were obtained for categorical variables. Hence, those values along
with empty cells and cells with value ‘NA’ (if any) were replaced with the value ‘nan’.

	survival	age	fractionalshortenin	epss	lvdd	w_score	w_index
count	133	133	133	133	133	133	133
mean	22.55	62.65	0.21	12.02	4.75	14.51	1.35
std	15.478	8.024	0.103	6.91	0.77	4.80	0.43
min	0.03	35	0.01	0	2.32	5	1
25%	10	58	0.15	7.6	4.29	11.67	1
50%	24	62	0.205	11	4.65	14	1.2
75%	33	67	0.26	14.8	5.25	16	1.5
max	57	86	0.61	40	6.78	39	3

Table 4.8: Descriptive statistics

Also, categorical variable, ‘group’ with missing value percent of 23 percent should be treated with necessary imputation. However, the data description obtained from UCI repository states that, variable ‘group’, ‘mult’ and ‘name’ can be ignored and ‘aliveat1’ is related to the variable ‘alive’ and hence these are dropped form the dataset. Median Imputation is carried out on numerical variables to avoid the impact of outliers on the same whereas Mode imputation on categorical variables.

	Missing	Percent
aliveat1	59	0.443609
group	23	0.172932
epss	16	0.120301
lvdd	12	0.090226
fractionalshortening	9	0.067669
age	8	0.06015
mult	6	0.045113
w_index	6	0.045113
w_score	6	0.045113
survival	5	0.037594
name	2	0.015038
pericardialeffusion	2	0.015038
alive	2	0.015038

Table 4.9: Missing Value Analysis Report

Histogram plots are obtained for numerical variables and is observed that, variable 'age' and 'lvdd' follows normality with higher kurtosis. Rest of the variables are not distributed normally and are positively skewed problem of higher kurtosis (figure 4.8). Transformation is carried out using 'Standardization' method on these numerical variables and this attempt failed to fit data into normal distribution (figure 4.9). Thus, Non-parametric statistical tests are carried out to obtain correlation matrix.

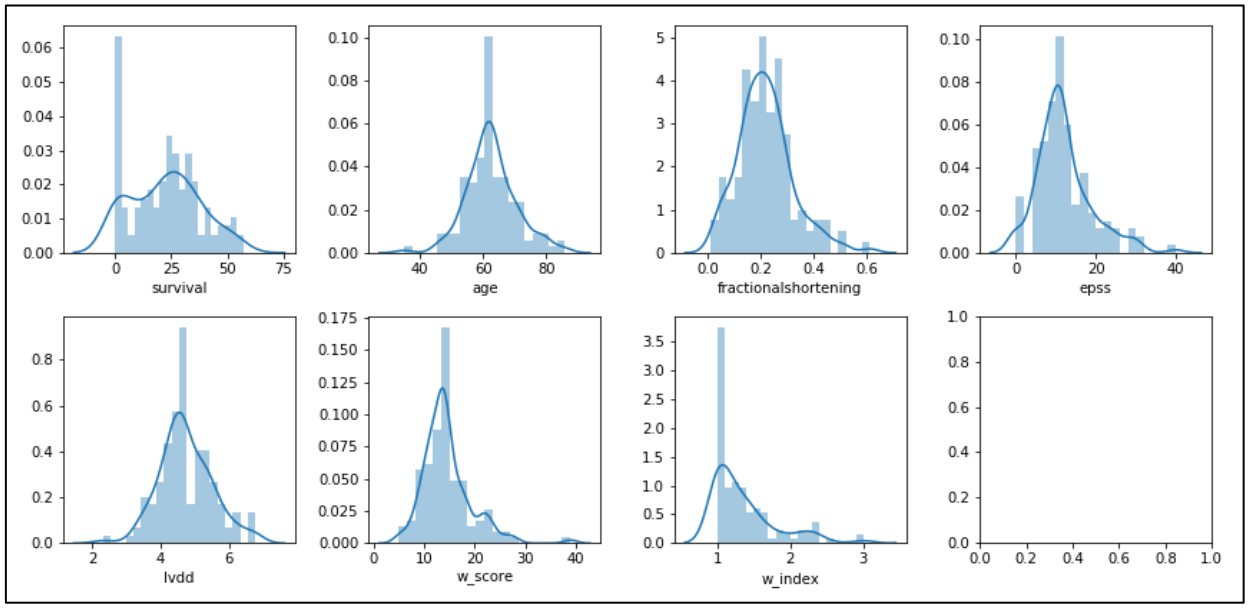


Figure 4.8: Histogram plots obtained before Normalization – Normality check

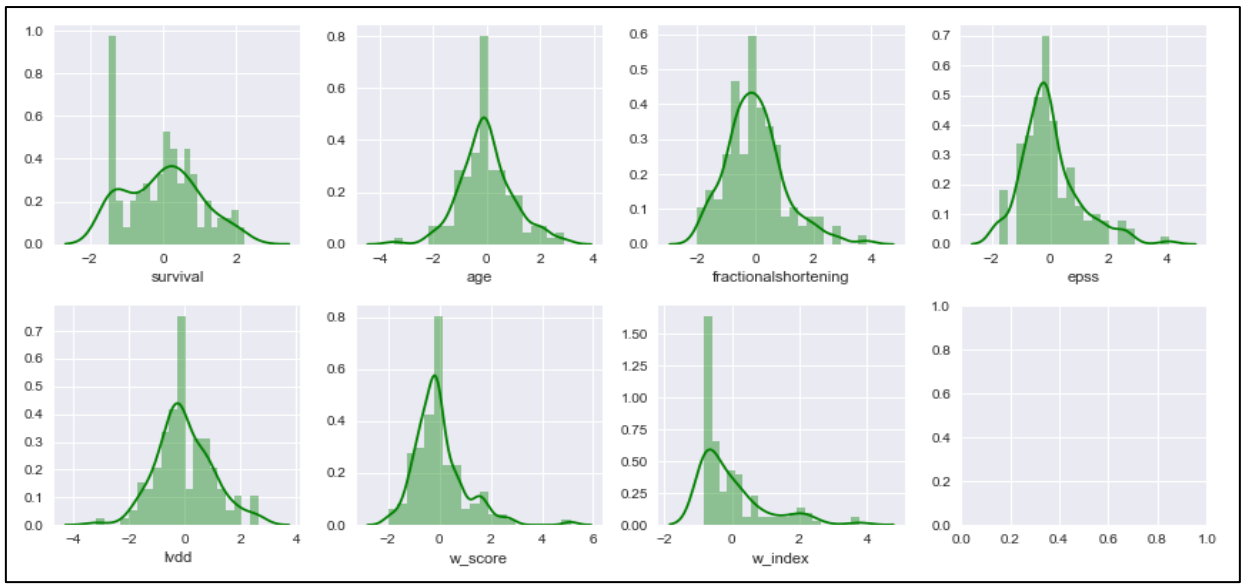


Figure 4.9: Histogram plots obtained after Normalization – Normality check

From figure 4.10, it is observed that there exists a class imbalance in the target variable ‘alive’. Because of which there is no clera representation of the influence of variable ‘pericardialeffusion’ on target variable. If the fluid is presentaround the heart, then there is a 50-50 chances of being alive or dead.

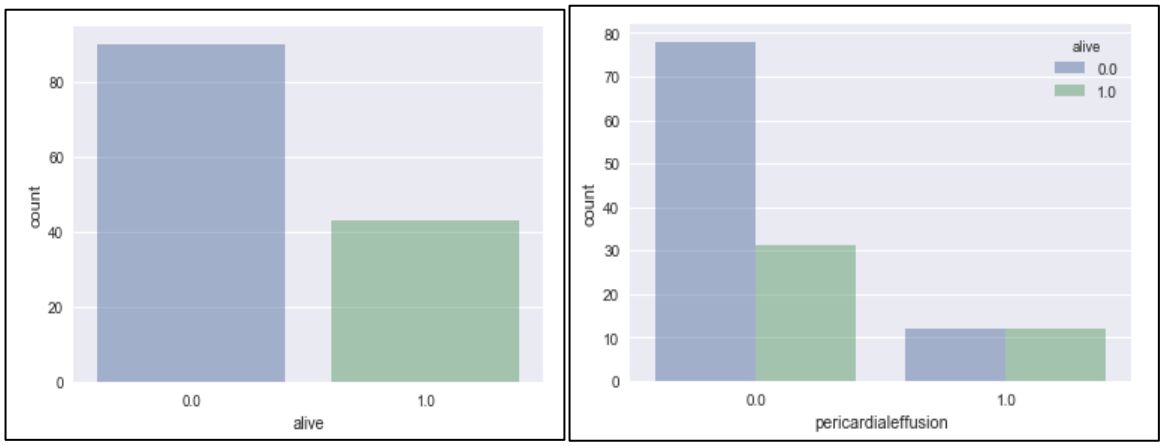


Figure 4.10: Frequency plot- Relation between Categorical and target variable

Non-parametric test Spearman Correlation is carried out to understand the relationship between independent variables and the target since the data is non-normally distributed. It is represented through a heat map which describes the correlation with correlation coefficient values. Correlation coefficients ranging from 0.1 to 0.3 is considered weak, 0.3 to 0.5 as moderate and 0.5 to 0.8 as strong in describing the relationship between variables. '+' indicates positive correlation and '-' indicates negative correlation. The results obtained are as briefed below,

1. A strong negative correlation exists between target variable 'alive' and survival.
2. A strong positive correlation exists between the variables epss and lvdd.
3. A strong positive correlation is found between independent variables w_score and w_index.
4. A negative moderate relationship is observed between variable survival, epss and w_index.
5. A moderate positive correlation exist between target variable, epss and w_index.
6. Variable fractionalshortenening share a moderate negative correlation with epss and lvdd.

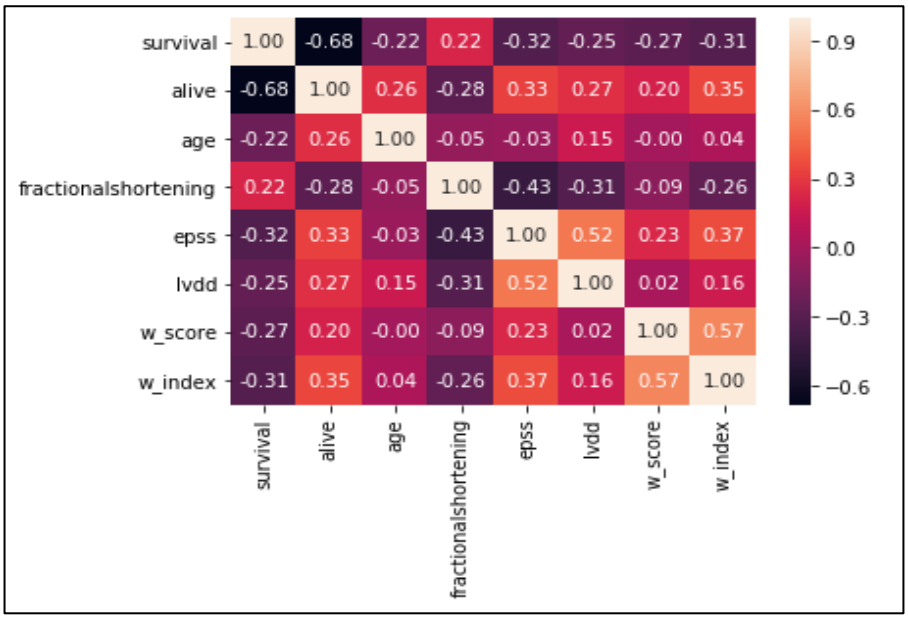


Figure 4.11: Correlation matrix represented in a heat map.

4.1.4 Immunotherapy

Descriptive statistics obtained for the dataset is as shown in table 4.10. Results generated from the missing value analysis had no traces of presence of missing values in the dataset and hence the data is clean (table 4.11).

	age	Time	Number_of_Warts	Area	induration_diameter
count	90	90	90	90	90
mean	31.04	7.23	6.14	95.7	14.33
std	12.23	3.09	4.21	136.61	17.21
min	15	1	1	6	2
25%	20.25	5	2	35.5	5
50%	28.5	7.75	6	53	7
75%	41.75	9.93	8.75	80.75	9
max	56	12	19	900	70

Table 4.10.: Descriptive statistics

	Missing	Percent
Result_of_Treatment	0	0
induration_diameter	0	0
Area	0	0
Type	0	0
Number_of_Warts	0	0
Time	0	0
age	0	0
sex	0	0

Table 4.11: Missing value analysis report

It can be inferred form the histogram plots obtained for numerical variables, that the data is not normally distributed, positive and negative skewness is found (fig. 4.12). Normalisation is carried out by transforming each numerical variables into its Z-scores. The outcome is as shown in the figure 4.13 and still the data follws non-normal distribution. Henceforth, Non-parametric statistical test is carried out to obtain the correlation matrix.

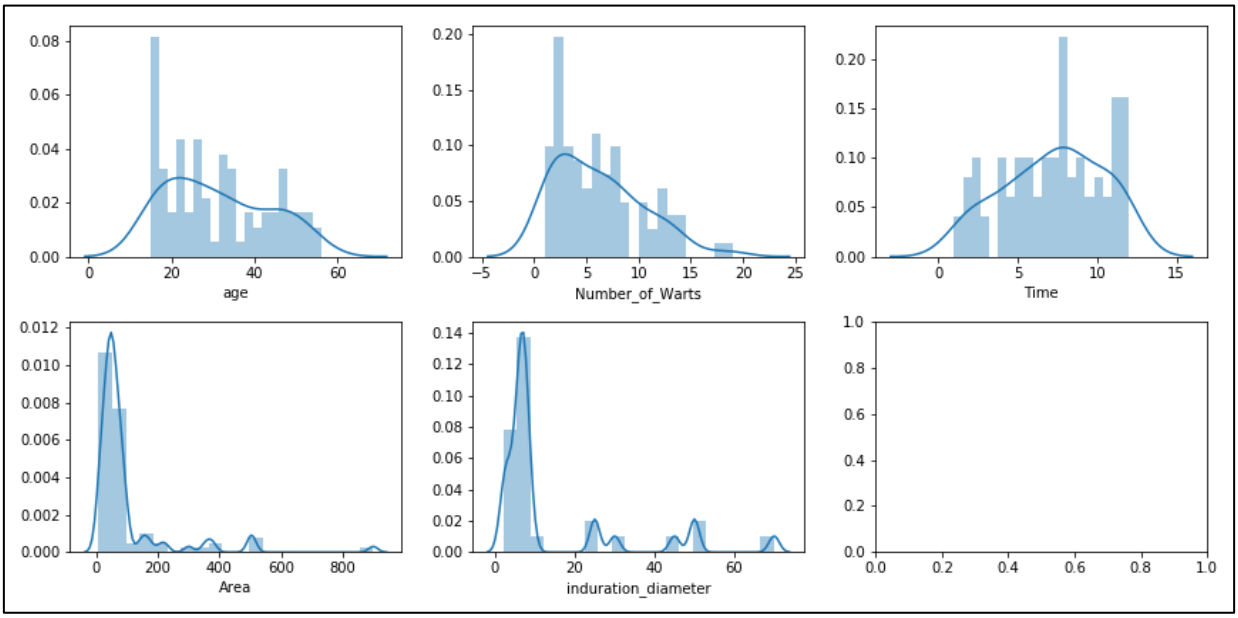


Figure 4.12: Histogram plots obtained before Normalization – Normality check

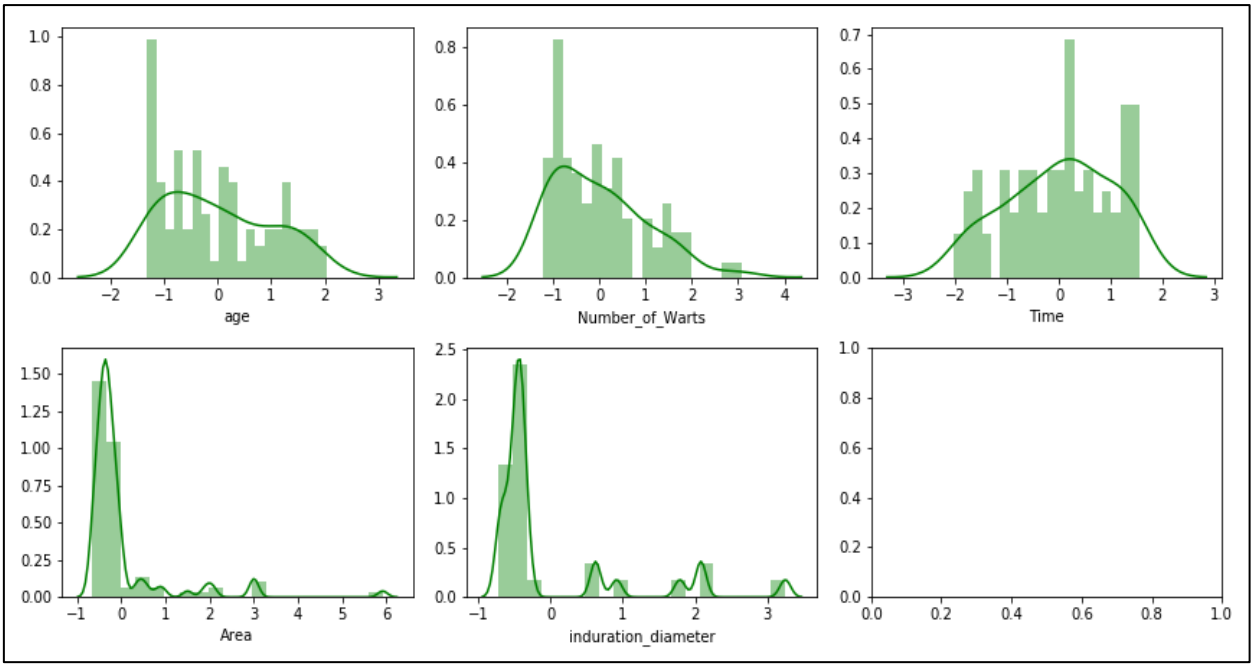


Figure 4.13: Histogram plots obtained after Normalization – Normality check

Figure 4.14 shows the relationship between categorical variables and the target. The target variable 'Result_of_treatment' shows class imbalance between its categories where 0 indicates the patient's negative response to the treatment whereas 1 indicates the opposite. Types of warts is categorized into type 1(plantar), type 2(common) and type 3(both) and there seems to be a relationship between the variable type and target.

Since dataset is not normally distributed, Spearman correlation a non-parametric statistical test is carried out to find the realtionship between target , the independent variables and itself. From the heat map (fig. 4.15), it can be infered that, a negative moderate correlation exists between the target and the variable 'Time' and the rest share weak correlation.

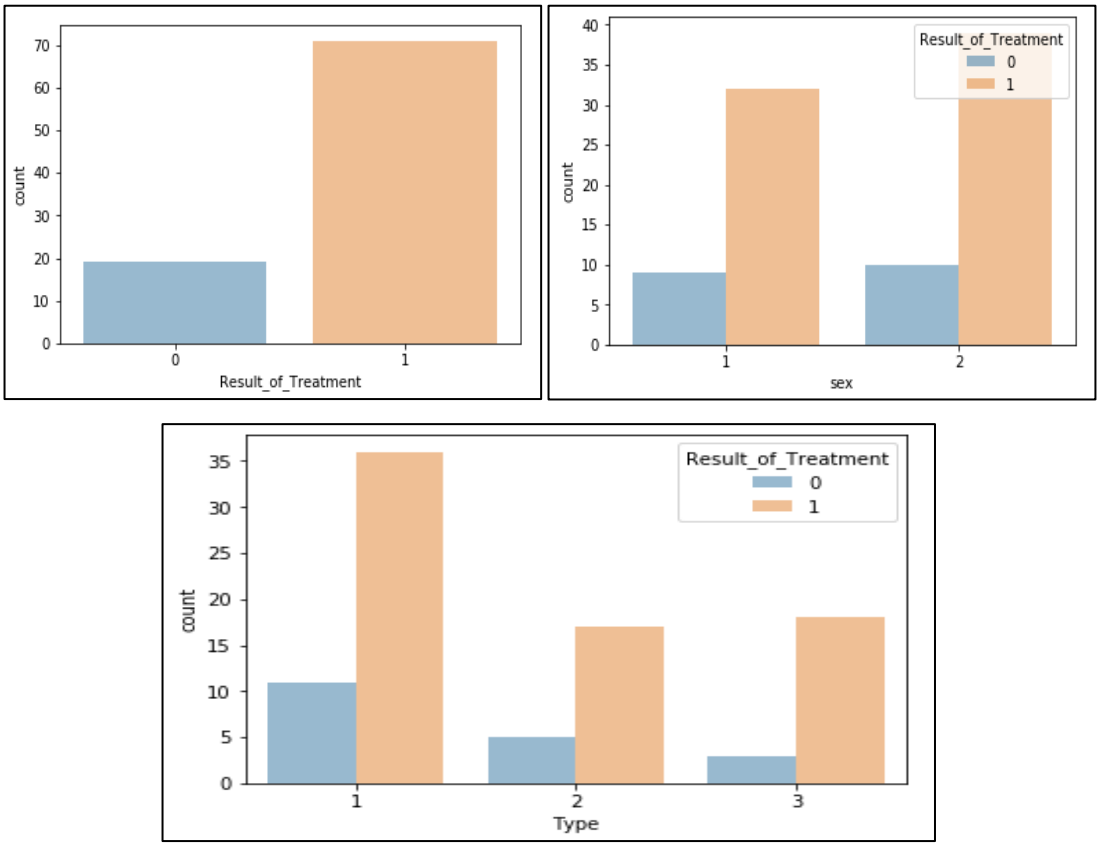


Figure 4.14: Frequency plot- Relation between Categorical and target variable

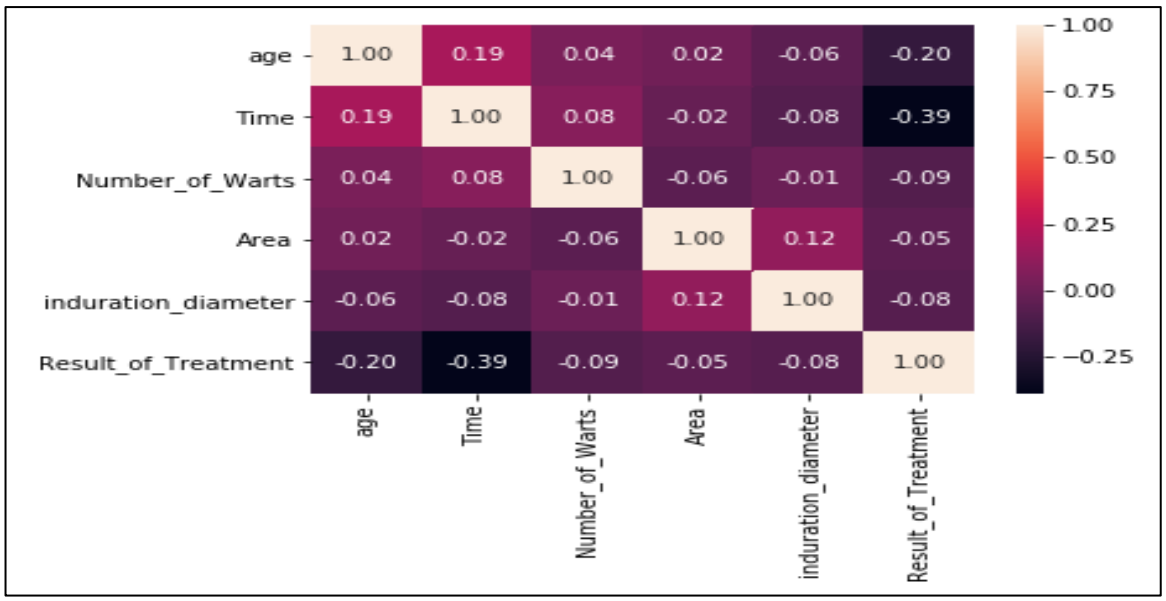


Figure 4.15: Correlation matrix represented in a heat map.

4.2 Data Preprocessing and Modelling

From the insight obtained in the previous section, suitable pre-processing techniques are applied on datasets, modeled and the results are documented.

Dataset is partitioned into 75 percent of training data and 35 percent of test data with a stratification of class variable to maintain the same imbalance ratio in test and train. The design as shown in figure 3.2 is ran for 5 times so that the data partition in each iteration will be random. Furthermore, Standardization of numerical variables is carried out on training data to scale all the values into common range. Dummy variables are created for each categorical variable in the dataset using the available function get_dummies in python Scikit-learn machine learning library. Similarly, target variables are label encoded into 0 and 1 using LabelEncoder module available in the same library.

Once the initial data cleaning is completed, various oversampling-methods are carried out on the same which includes SMOTE, SMOTE-Borderline, ADASYN and SIMO. Equal amount (1:1) of minority and majority instances were obtained in resampled balanced dataset on the application of SMOTE, SMOTE- Borderline 1 and SMOTE-Borderline 2 whereas the ratio of minority class obtained was higher than the majority class

instances in the new resampled balanced dataset obtained on the application of ADASYN. Since our major focus is on SIMO and as mentioned in the literature that less synthetics instances will be generated as the motive is to oversample informative minority instances alone, following is tabulated for observation.

Dataset: Biomarker

Iterations	SIMO parameters	SIMO, Balance ratio
1	delta = 0.8, p = 0.6	0: 43, 1: 28
2	delta = 0.8, p = 0.6	0: 43, 1: 52
3	delta = 0.8, p = 0.6	0: 43, 1: 47
4	delta = 0.8, p = 0.6	0: 43, 1: 35
5	delta = 0.8, p = 0.6	0: 43, 1: 15

Table 4.12: Details on SIMO parameter settings and the resampled ratio

Dataset: Hepatitis

Iterations	SIMO parameters	SIMO, Balance ratio
1	delta = 0.4, p = 0.5	0:60, 1:73
2	delta = 0.4, p = 0.5	0:60, 1:73
3	delta = 0.4, p = 0.5	0:60, 1:73
4	delta = 0.4, p = 0.5	0:60, 1:73
5	delta = 0.4, p = 0.5	0:60, 1:73

Table 4.13: Details on SIMO parameter settings and the resampled ratio

Dataset: Echocardiogram

Iterations	SIMO parameters	SIMO, Balance ratio
1	delta = 0.2, p = 0.25	0:53, 1:34
2	delta = 0.2, p = 0.25	0:53, 1:48
3	delta = 0.2, p = 0.25	0:53, 1:40
4	delta = 0.2, p = 0.25	0:53, 1:41
5	delta = 0.2, p = 0.25	0:53, 1:40

Table 4.14: Details on SIMO parameter settings and the resampled ratio

Dataset: Immunotherapy

Iterations	SIMO parameters	SIMO, Balance ratio
1	delta = 0.8, p = 0.6	0:46, 1:42
2	delta = 0.8, p = 0.6	0:18, 1:42
3	delta = 0.8, p = 0.6	0:47, 1:42
4	delta = 0.8, p = 0.6	0:46, 1:42
5	delta = 0.8, p = 0.6	0:29, 1:42

Table 4.15: Details on SIMO parameter settings and the resampled ratio

It can be inferred from the tables that the synthetic data generated at each iteration varied and with respect to balanced ratio achieved, generated minority points were either half or less the majority points (Highlighted cells in the table) or higher than majority points present in the train set. Also, the obtained number of resampled records is less than induced training set as in the SIMO approach, training data is again split into 80 percent of training and 20 percent of validation set. Thus, oversampling is carried out only on the split 80 percent of train set.

SIMO parameters, 'delta' is used to choose percentage of informative minority data examples located at the boundary of the SVM for oversampling and 'p' is used to regulate the amount of synthetic data to be generated by SIMO on oversampling. Sample size of the chosen datasets had an impact on these parameters where the small datasets demanded higher delta and p value to perform oversampling. An error example found during execution is shown below,

***Error**: Expected n_neighbors <= n_samples, but n_samples = 1, n_neighbors = 6*

Meaning, there is no sufficient neighbors to generate synthetic examples by KNN.

Thus, values to the parameters was increased in steps of 0.5 and was continued till there was no error thrown by SIMO. The discussed problem was found while working on datasets Biomarker and Immunotherapy or records 90.

The proposed design in the experiment is executed five times to thoroughly evaluate classifier's performance. The base classifiers, Logistic regression and SVM is used to have a simple approach in investigating the performance of imbalanced learning technique SIMO on small datasets. GridSearchCV along with the specified supervising learning algorithms is used to improve the performance of the classifiers by selecting best hyperparameters out of given ones. A range of values selected for tuning parameters 'C', 'penalty' and 'gamma' is as shown in the table below.

	Logistic Regression
C	0.001, 0.01, 0.1, 1, 10, 100, 1000
Penalty	'l1' and 'l2'

Table 4.16: Tuning parameters for Logistic Regression

	SVM
C	1, 2, 3, 4, 5, 6,7, 8, 9, 10
gamma	0.01, 0.02, 0.03, 0.04, 0.05, 0.10, 0.2, 0.3,0.4, 0.5

Table 4.17.: Tuning parameters for SVM

4.3. Results and Discussions

4.3.1 Results

Evaluation of the models is carried out in terms of performance metrices obtained from confusion matrix; Accuracy, Precision, Recall, F-score, G-mean and ROC-AUC plots. Since the chosen datasets are facing class imbalance problem, Accuracy, G-mean and ROC plots are carefully observed for performance comparison. Consideration wise, Accuracy comes the last. The metrices are obtained when data is modelled using classifiers Logistic regression and Support Vector Machine.

Logistic regression and SVM is trained on both balanced and imbalanced sets and the results are obtained on test set. The design is iterated for 5 times and the table with

average values of Accuracy, G-mean and AUC is generated. To make evaluation easier graphs are obtained for the same.

1. Dataset: Biomarker

From the figure 4.16 and 4.17, it can be reported that, models performed well on the application of approaches SMOTE, SMOTE-Borderline and ADASYN when compared to SIMO. Models obtained 100 percent accuracy on both balanced and imbalanced train sets. SIMO fetched good results for Logistic regression (92 percent AUC and 91 percent G-mean) and moderate for SVM (76 percent G-mean and 77 percent AUC).

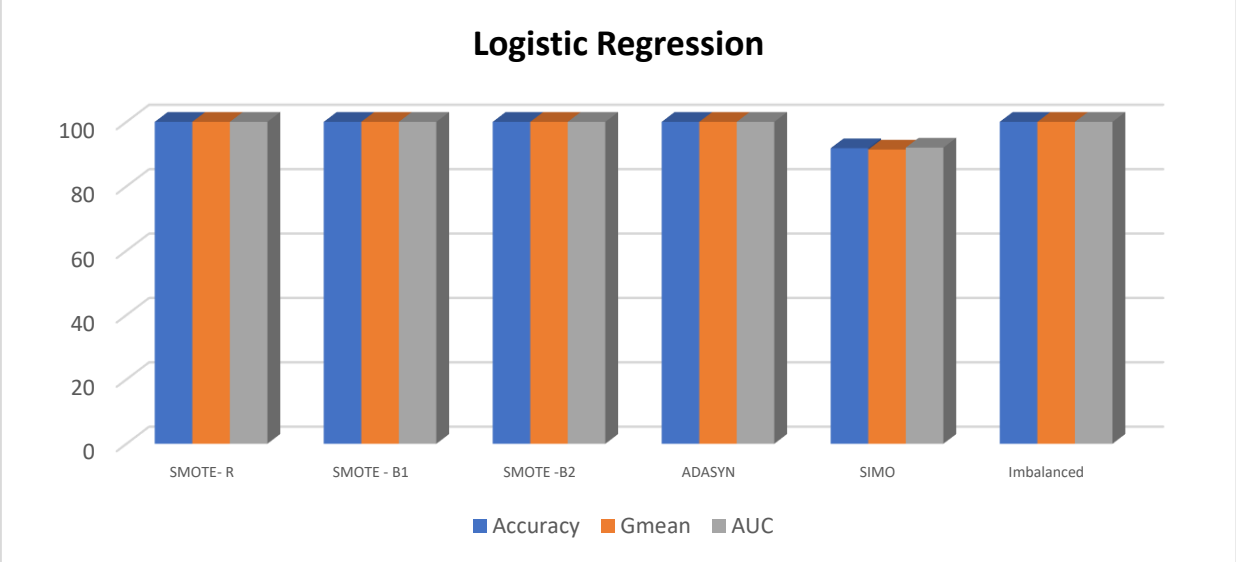


Figure 4.16: Performance Comparison bar graph: Biomarker – Logistic regression

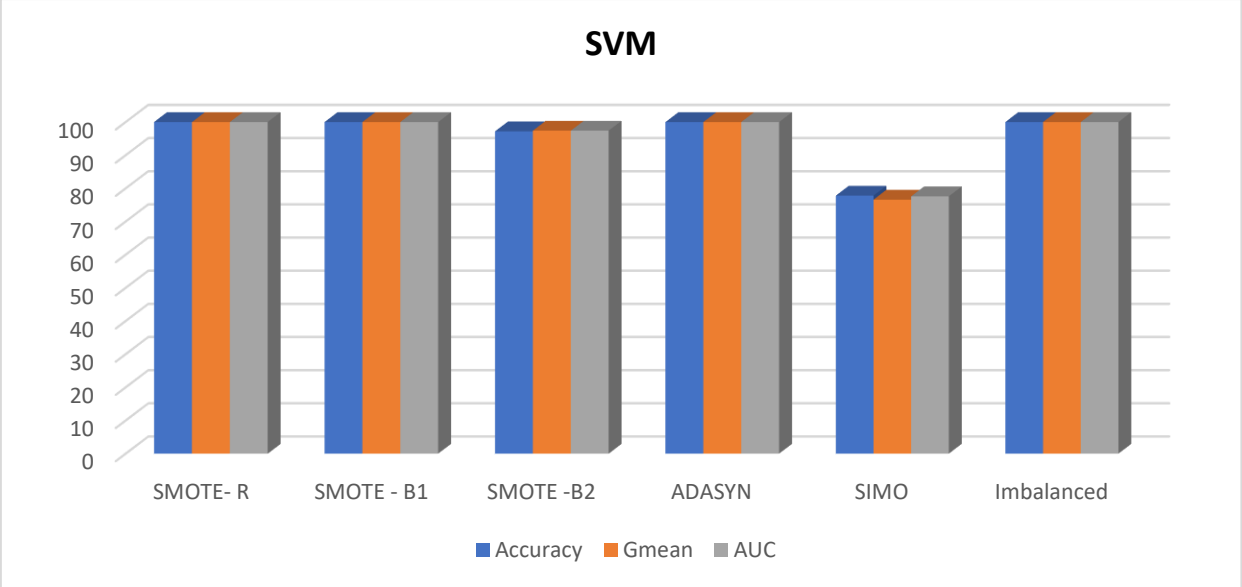


Figure 4.17: Performance Comparison bar graph: Biomarker - SVM

2. Dataset: Hepatitis

Accuracies obtained for the models trained on data oversampled by each approach is almost the same (figure 4.18). But, difference between accuracy and other two metrices G-mean and AUC does exist, largely in the case of SVM. Models performances on the application of baseline imbalance learning approaches are slightly similar and better than SIMO (70 percent – G-mean, 72 percent – AUC).

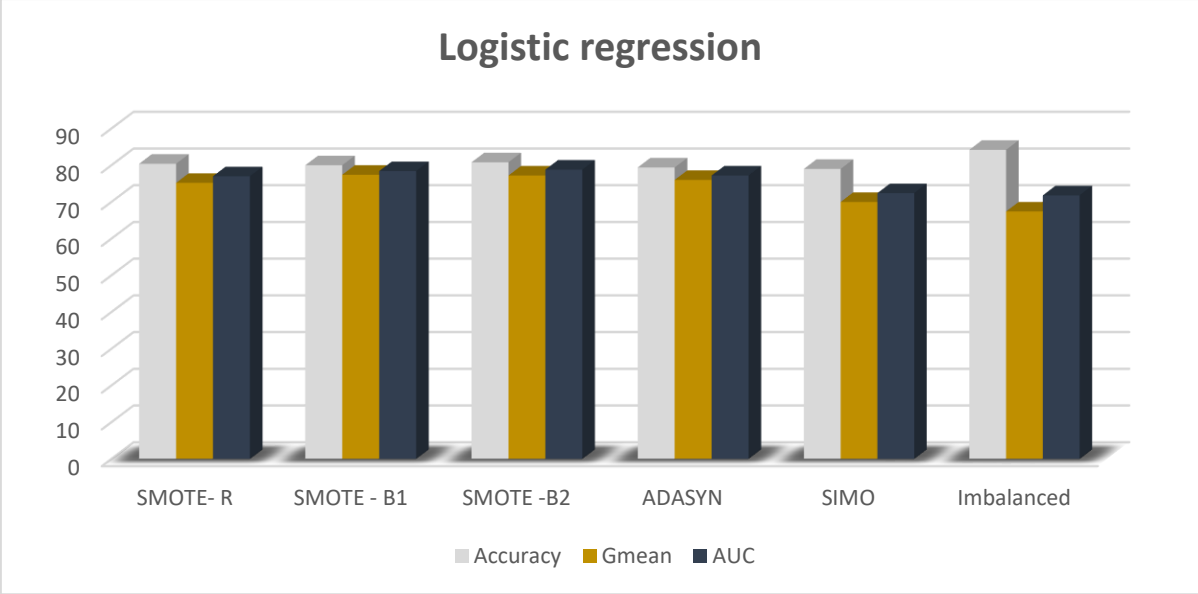


Figure 4.18: Performance Comparison bar graph: Hepatitis – Logistic Regression

However, approach SIMO outperformed the other baseline learning algorithms on modelling resampled data set using SVM (figure 4.19). SMOTE-R and ADASYN has performed the worst in terms of G-mean and AUC.

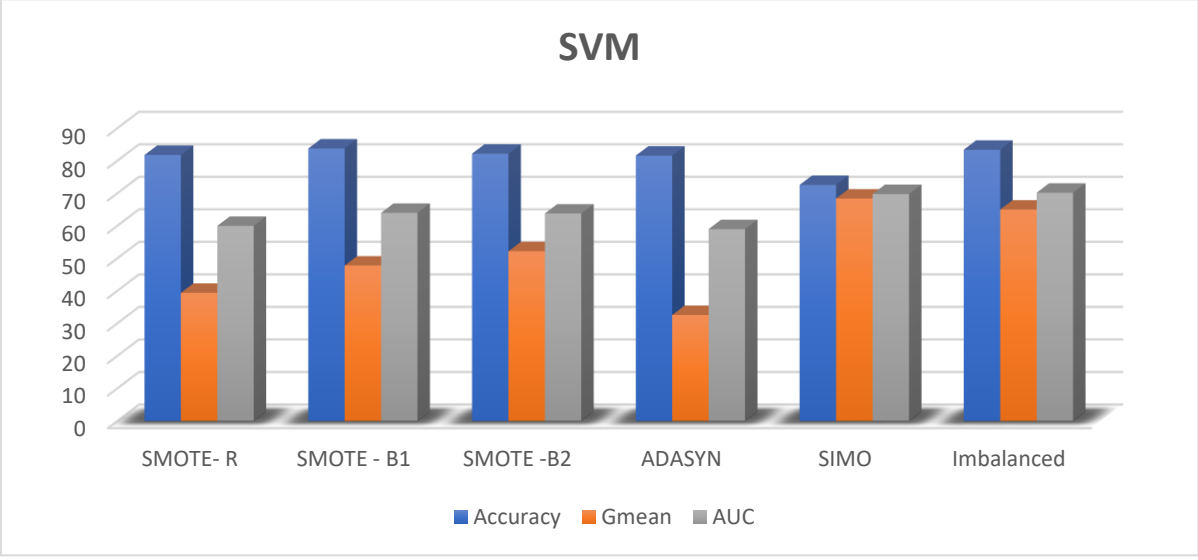


Figure 4.19: Performance Comparison bar graph: Hepatitis - SVM

3. Dataset: Echocardiogram

From figure 4.20, performance of Logistic regression scored highest with the approach SMOTE, followed by SIMO and ADASYN performing equally (G-mean: 81 percent and AUC: 82 percent).

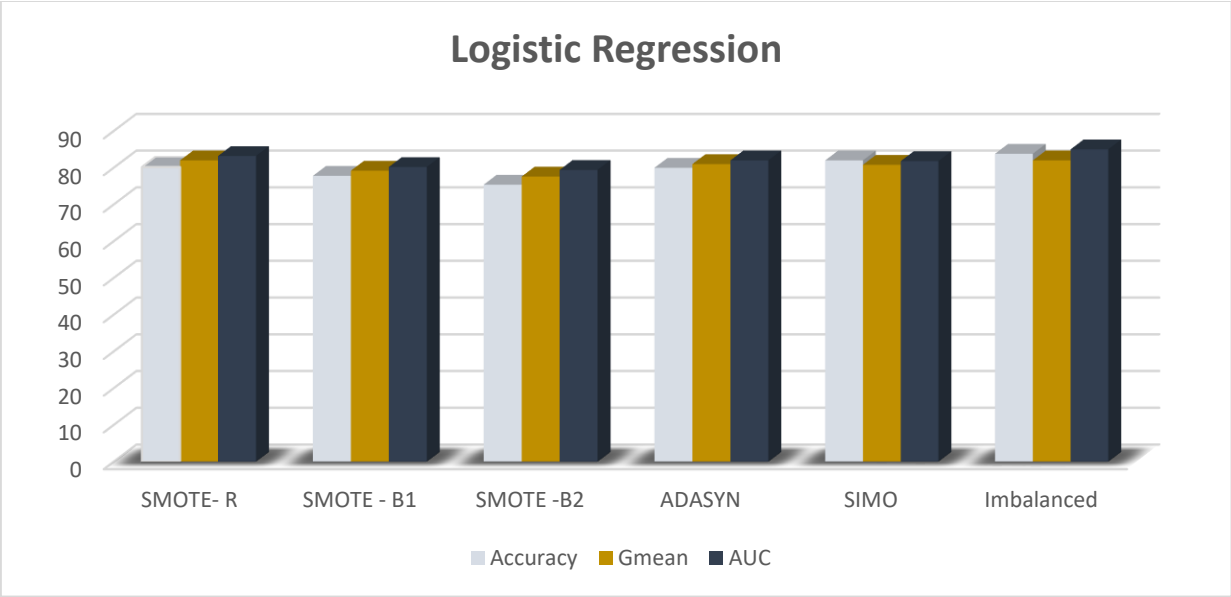


Figure 4.20: Performance Comparison bar graph: Echocardiogram – Logistic Regression

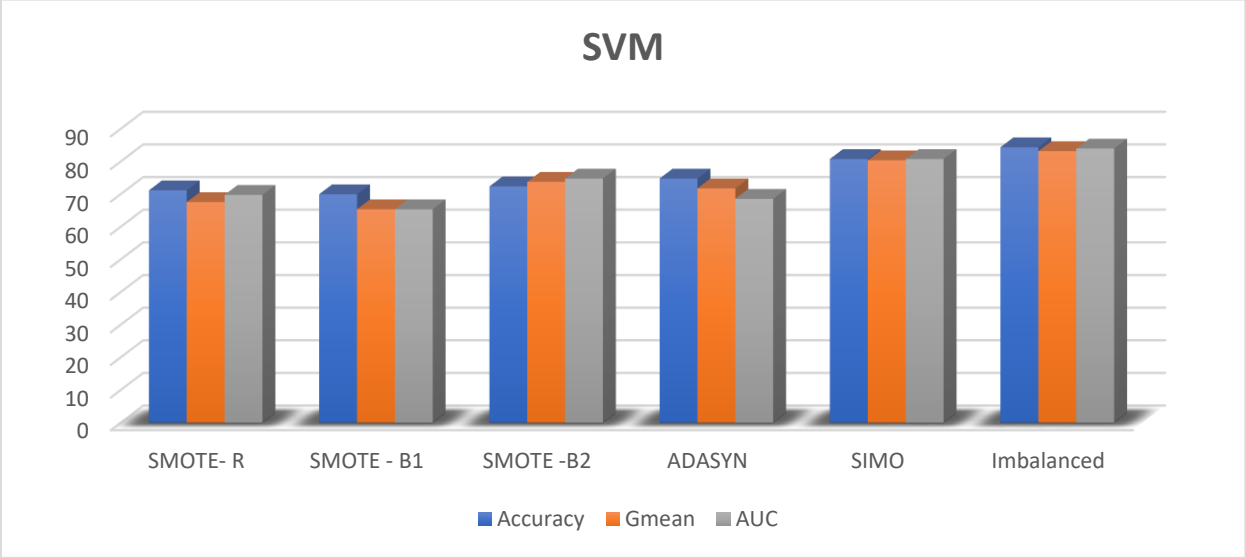


Figure 4.21: Performance Comparison bar graph: Echocardiogram - SVM

SIMO outdid the other approaches when trained with SVM with a G-mean, AUC and Accuracy of 81 percent. SMOTE-Borderline 2 scored the second in performance (figure 4.21).

3. Dataset: Immunotherapy

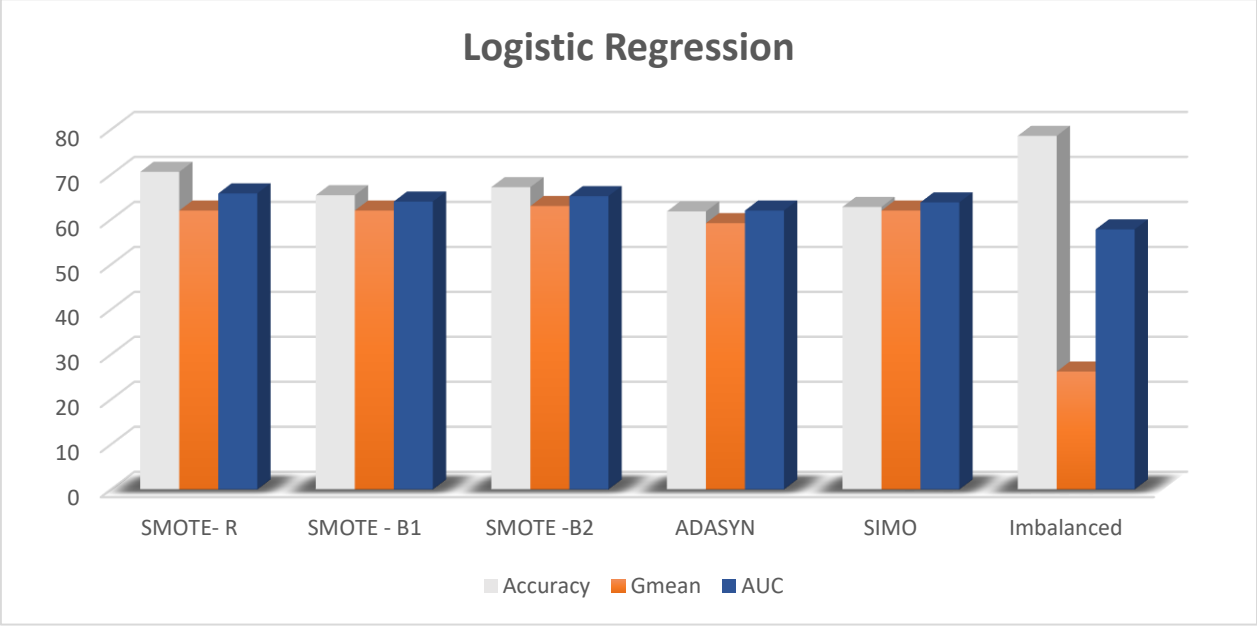


Figure 4.22: Performance Comparison bar graph: Immunotherapy – Logistic regression

Logistic regression models build on the train set resampled by the approaches SIMO, SMOTE, SMOTE- Borderline1 and SMOTE-Borderline 2 has equally performed in terms of G-mean (62 percent) with a slight difference in their AUC values (figure 4.22). However, on application of SIMO, SVM has performed well when compared to other models (figure 4.23). Other models have performed worst with respect to G-mean. Neither of the models performed better on this dataset as there were instances of zero G mean and 50 percent of AUC in the performed iterations.

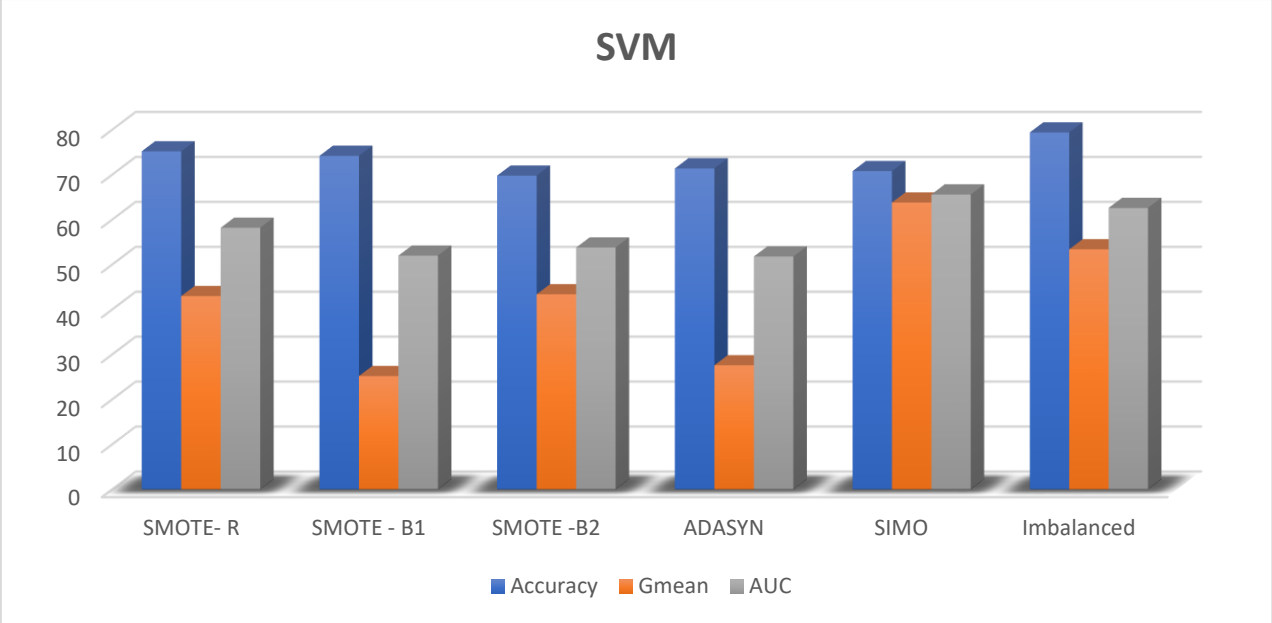


Figure 4.23: Performance Comparison bar graph: Immunotherapy - SVM

4.3.2 Statistical Significance and Hypothesis Evaluation

`Wilcoxon Signed-Rank Test', a statistical test is carried out to check if the results obtained are statistically significant or not with the chosen cut-off value p = 0.05. This is performed in support of rejecting or retaining the stated hypotheses stated below,

Hypothesis:

H_0: "The G-mean and AUC values of the models built on the oversampled datasets using imbalanced learning algorithm SIMO is equal to the G-mean and AUC values obtained from the models on application of baseline algorithms SMOTE, SMOTE-Borderline and ADASYN, with p-value < 0.05."

** *AUC – Area Under Curve*

** *SIMO - Synthetic Informative Minority Over-Sampling leveraging SVM*

** *SMOTE - Synthetic Minority Over-Sampling Technique*

** *ADASYN - Adaptive Synthetic Sampling Approach*

The results are obtained by comparing the G mean and AUC values of SIMO with values of other approaches obtained from Logistic regression and SVM models in each iteration.

From table 4.18,

1. Hypothesis is rejected for dataset biomarker, meaning there is a statistically significant difference in the performance of the models on the application of SIMO over the other baseline approaches ($p < 0.01$). But, the difference is the performance of the SIMO is poorer than others.
2. Hypothesis is true and can be retained for model Logistic regression in all the datasets except for Biomarker. There is no statistically significant difference found between the logistic regression models trained on different approaches.
3. Hypothesis is false and can be rejected for SVM models in the dataset hepatitis, as there is a statistically significant difference between the performance of the imbalanced learning algorithms. ($p < 0.01$)
4. In Echocardiogram dataset, there is no statistically significant difference between SMOTE-Borderline 2 and SIMO ($p = 0.54$ for G mean and $p = 0.45$ for AUC). Meaning there is no performance difference.
5. In the dataset Immunotherapy, statistically significant difference doesn't exist between the algorithms SIMO and baseline algorithms SMOTE with respect to AUC ($p= 0.139$) and SMOTE-Borederline2 with respect to G mean ($p = 0.07$) and hence hypothesis can be retained.

Dataset	LR				SVM			
BIOMARKER	G mean		AUC		G mean		AUC	
	t-statistics	p	t-statistics	p	t-statistics	p	t-statistics	p
SMOTE	2	0.0015	1	0.0012	0	0.0007	0	0.0006
SMOTE_B1	2	0.0015	1	0.0012	0	0.0007	0	0.0006
SMOTE_B2	0	0.001	0	0.001	14	0.0089	1	0.0012
ADASYN	3	0.0019	1	0.0012	0	0.0007	0	0.0006
IMMUNOTHERAPY	G mean		AUC		G mean		AUC	
	t-statistics	p	t-statistics	p	t-statistics	p	t-statistics	p
SMOTE	41	0.47	49	0.8259	22	0.055	34	0.139
SMOTE_B1	37	0.191	44.5	0.3781	18	0.017	17.5	0.0157
SMOTE_B2	27	0.06	32	0.1115	28.5	0.0732	20	0.022
ADASYN	51.5	0.629	57.5	0.8869	13.5	0.0143	14	0.0089
HEPATITIS	G mean		AUC		G mean		AUC	
	t-statistics	p	t-statistics	p	t-statistics	p	t-statistics	p
SMOTE	41	0.75	47	0.7297	5	0.0017	12.5	0.0069
SMOTE_B1	47.5	0.75	50	0.8751	1	0.0008	10.5	0.0049
SMOTE_B2	57	0.86	57.5	0.887	1	0.0008	7	0.0026
ADASYN	57	0.86	55.5	0.7981	8	0.0031	18	0.017
ECHOCARDIOGRAM	G mean		AUC		G mean		AUC	
	t-statistics	p	t-statistics	p	t-statistics	p	t-statistics	p
SMOTE	33	0.3812	32.5	0.6052	5	0.0046	6	0.0056
SMOTE_B1	31	0.31	45	0.971	18.5	0.0184	17.5	0.015
SMOTE_B2	59	0.9546	48	0.777	49.5	0.549	40.5	0.4495
ADASYN	49.5	0.85	48.5	0.8012	0	0.001	0	0.001

Table 4.18: Wilcoxon Signed-Rank Test results

4.3.3 Discussions

The study's key focus was to validate the performance of novel imbalance learning algorithm SIMO on small datasets. From the results obtained it can be concluded that, training Logistic regression on train resampled by SIMO fetched bad results for dataset 'Biomarker', moderate metrices values for datasets; 'Immunotherapy' and 'Hepatitis' and

better for dataset ‘Echocardiogram’. Also, SVM trained on resampled train sets by SIMO outperformed baseline imbalance learning algorithms in all the datasets except for ‘Biomarker’.

The results obtained for Biomarker by the application of baseline algorithms are brilliant when compared to SIMO, yet suspicious to overfitting as the average AUC, Accuracy and G-mean values obtained are 100 percent. On the other edge, SIMO’s performance is moderate scoring average of approximately 92 percent when trained with logistic regression and approximately 78 percent with SVM. These two scores seem quite moderate and acceptable. On the similar note, either there existed an overfitting or the SIMO performed bad.

Dataset ‘Biomarker and Immunotherapy’ are of the records 93 and 90 with an imbalance ratio are 60:40 and 80:20 respectively. The SIMO parameters ‘delta’ and ‘p’ is set to 0.8 and 0.6 respectively for these two datasets which are of higher than those suggested in the literature. Factors of the dataset that are influencing the parameters of SIMO is unknown. However, on careful observation it is found, the dataset ‘Hepatitis’ of record 154 sharing the same imbalance ratio as ‘Immunotherapy’, performed well with ‘delta’ value set to 0.4 which signifies the role of sample size on ‘delta’. This is in contrary to what the author had observed while experimenting on datasets of range 300 to 1500 and concluded reporting, ‘SIMO is not very sensitive to the value of its parameters’. (Piri, Delen & Liu, 2018)

‘Biomarker’ dataset, though its imbalance ratio is 60:40 which is moderate, demanded higher values of delta and p. SIMO algorithm threw an error of insufficient data availability to generate the required synthetic examples during the experiment. This is because SIMO adopts SMOTE approach and on inadequate availability of informative minority points (minority points in general) to generate synthetic data the technique will fail to oversample. Hence, on setting higher values for ‘delta’, larger number of minority examples will be selected and oversampled using SMOTE approach. Increase in ‘p’ will increase the iterations but there will be a possibility of picking more relevant informative

examples located near decision boundary on every application of SVM on the updated sample in the loop till the imbalanced gap is reached (SIMO Algorithm). Thus, both the sample size and class imbalance ratio will have an influence on determining the parameters for SIMO. This can be supported by the dataset 'Echocardiogram' of records 131 and imbalance ratio of 70:30 performing well at 'delta' set to 0.2 and 'p' to 0.25.

Feature selection isn't used in the experiment as data is very small and attributes are of acceptable dimension. This could be a possible reason for the suspicious performance results obtained for dataset 'Biomarker' on the application of baseline imbalanced learning algorithms, hinting the presence of overfitting. Models performed moderate yet poorly on dataset 'Immunotherapy' which needs further investigation.

In a nutshell, SVM models trained on data resampled by SIMO fetched satisfactory results as the algorithm can be better or even worse based on the size and complexity of the sample. The approach either outperformed or performed equivalent to other approaches. Dataset Biomarker is exceptional. The use of datasets is limited to four because of time and data availability constraints. From the obtained fluctuating results, it is hard to comment on the efficiency of the algorithm as it performed moderate yet better than all other approaches for 'Immunotherapy' and performed bad for 'Biomarker' compared to all other approaches. Thus, considering more datasets will be convenient in deriving a conclusion.

Evaluation of the models involved analyzing performance metrices obtained from the models. These were obtained from Confusion matrix. Accuracy isn't given much importance in this experiment because in class imbalance problem, classifiers tend to misclassify minority into majority and end up giving higher accuracy. Thus, G-mean and AUC values are taken into consideration. AUC results are fetched from ROC plots. In support of the explanation, it can be observed from the graphs that, AUC and G-mean values closely followed each other, meaning they had similar values or with slightly different values. While, the difference between Accuracy and the duo (AUC and G-mean) was higher and quite noticeable.

Strengths

1. Avoidance of overfitting.
2. Adaptability to any range of small datasets and ratio of class imbalance due to the availability of parameter setting options ‘delta’ and ‘p’.
3. Oversampling technique with an advantage of SVM in selecting informative minority instances available near decision boundary and G-mean in retrieving the fine oversampled dataset.
4. Performed better with small datasets of high class imbalance ratio with supervised learning algorithm Support Vector Machine.
5. Except for biomarkers, satisfactory results were obtained when compared to SMOTE, SMOTE-Borderline and ADASYN.

Limitations

1. Use of four datasets to evaluate the efficiency of the algorithm.
2. Other machine learning algorithms like decision tree, random forest is not used in this experiment which would have given the broader understanding on the behavior of SIMO on these classifiers.
3. Synthetic data generated was not always lesser and computational training time wasn’t less as mentioned in the literature (Piri, Delen & Liu, 2018).

CHAPTER 5

CONCLUSION

5.1 Research Overview

The research is carried out to analyze the impact of novel imbalanced learning technique 'Synthetic informative minority over-sampling (SIMO) algorithm leveraging support vector machine' on the performance of classifiers when applied on very small dataset. Initially the research is started with a literature review, discussing the available imbalanced learning algorithms, its merits and demerits over each other, problem of class imbalance and its impact on the classifiers. Also, the approach used in this experiment, is explained in detail along with its merits over other approaches.

Four small datasets of records less than 150 is collected and is analyzed and preprocessed by standardizing numerical variables and one-hot encoding categorical variables. As the dataset is very small train-test split of ratio 75:25 is obtained. To evaluate the performance of the SIMO on classifiers, baseline imbalance learning algorithms SMOTE, SMOTE-Borderline, ADASYN are also implemented in this study. Base classifiers Logistic regression and SVM-linear is used to keep the approach simple. GridSerachCv with 5-folds is used to choose hyperparameters for classifiers to improve their performance. The design is iterated for five times to investigate the performance of the models thoroughly for its reliability and stability. Performance of the models are evaluated using the metrices calculated from confusion matrix. G-mean, Accuracy and ROC plots are of the main focus to examine model's performance in this research. The average of metrices is calculated, tabulated and graphs are obtained to assess the performance difference of the learners on the application of SIMO and other oversampling techniques.

5.2. Problem Description

Developing predictive models for classification problems considering imbalanced datasets is one of the basic difficulties in data mining and decision-analytics. A classifier's performance will decline dramatically when applied to an imbalanced dataset. Standard

classifiers such as logistic regression, Support Vector Machine (SVM) are appropriate for balanced training sets whereas provides suboptimal classification results when used on unbalanced dataset. Performance metric with prediction accuracy encourages a bias towards the majority class, while the rare instances remain unknown though the model contributes a high overall precision. There are chances where minority instances might be treated as noise and vice versa. (Haixiang et al., 2017)

Class imbalance and small datasets are the problems which is dealt in this research and losing any piece of informative data isn't affordable. There are several oversampling methods available with their own benefits and limitations. On reviewing several literatures, it is found that either the technique will be prone to overfitting or overgeneralization. The benefits of SIMO include less computational training time and avoidance of overfitting with consideration of informative instances alone for sampling. Also, the approach outperformed the other approaches reviewed in the literature (like SMOTE, Smote-Borderline, Safe-level SMOTE) when applied on datasets of records in the range 300 to 1500 and hence this approach is implemented in the research to examine if the performance will be similar in case of small datasets with high class imbalance.

To investigate the same, following research question is posed,

"*Can performance metrices of the classifiers on small datasets, significantly improve on the application of 'SIMO leveraging SVM' over the application of baseline imbalanced learning algorithms?*"

***SIMO: Synthetic Informative Minority Over-Sampling*

***SVM: Support Vector Machine*

***Baseline imbalanced learning algorithms: SMOTE, SMOTE-Borderline1, SMOTE- Borderline2 and ADASYN.*

***Performance metrices: Accuracy, Recall, Precision, G-mean and ROC*

5.3 Contribution and Impact

To analyze the performance of the imbalanced learning technique SIMO on small dataset with class imbalance, dataset serving the same criteria is used. The records used are

no more than 150, similar patterns can be obtained in the domain of biomedical and related fields. Limited datasets were available for classification problems of this requirement, since our interest was on Univariate class examples alone, four datasets are finalized for this experiment.

Data is meticulously examined for missing and irrelevant values. The distribution of the data is observed carefully to decide on the imputation method to be used. Since the data was non-normally distributed, and as it followed the same distribution even after transformation, median imputation is carried out on numerical variables to avoid effect of outliers and mode imputation on categorical.

Dataset is partitioned into 75:25 train test sets instead of using cross-validation methods, to make enough data available for the classifiers in learning. Also stratified partitioning is carried out to avoid bias and overfitting. Standardized (Z-score) and one-hot encoded train set is oversampled using SMOTE, SMOTE-Borderline, ADASYN and SIMO. The aim is to compare the performance of SIMO over other mentioned algorithms when applied n small datasets. To keep the approach simple, base classifiers Logistic regression and SVM are chosen along with GridSearchCv (five folds) to obtain optimal model result.

On evaluation it is noticed that, Logistic regression trained on SIMO oversampled data fetched satisfactory results, however poor results were obtained for dataset 'biomarker' when compared to other models trained on of baseline imbalanced learning algorithms oversampled train set. SVM modelled on SIMO oversampled train data, outperformed all the other baseline algorithms except for dataset 'Biomarker'. The possible reason for the poor results when implemented on dataset 'Biomarker' is discussed in Results and Discussions section. In a nutshell, SIMO was consistent in its performance for all the range of datasets and imbalance ratio while other approaches failed to give better results.

5.4 Future Work and Recommendations

The aim of the study was to examine the performance of imbalanced learning algorithm, 'Synthetic informative minority over-sampling (SIMO) algorithm leveraging support vector machine' on small datasets. During the study the major limitation was dealing with the small dataset itself. As losing any information wasn't in the interest of this

experiment and due to time constraint, feature selection wasn't included in the design of the experiment. This can be considered as a future work in investigating the performance of the implemented algorithm on the use of feature selection. Also, as data is very small and again there will be data partition in SIMO, use of cross validation is avoided which can also be a part of future work.

The experiment is constrained to four datasets due to time constraint and the data availability. The experiment can be extended including more datasets of different sample size and class imbalance scenarios. This can be accompanied with having other supervised learning algorithms like random forest and decision tree to investigate SIMO performance on different classifiers. Modifications to the SIMO algorithm can be done by using Biased SVM instead SVM which might provide a different direction to the research. Idea of Biased-SVM is derived from (Hartono, Sitompul, Tulus & Nababan, E. ,2018) which also explains Cluster-SVM. On thorough research of mentioned topics, future research can be carried out on replacing SVM with Biased SVM if viable in contributing to the class imbalance field.

From the results obtained in this study, the novel imbalance learning technique 'Synthetic informative minority over-sampling (SIMO) algorithm leveraging support vector machine' cannot be recommended to oversample small datasets. The focus is on oversampling informative minority samples which are noise free thus eliminating the problem of overfitting. Parameter settings is an added advantage which is helpful in dealing with different small data samples and varied imbalance ratio. But, the results obtained are questionable for dataset Biomarker and Immunotherapy the datasets of records 90 which are comparably very smaller than the other two chosen in the experiment.

Since it is known that the algorithm performance varies accordingly with small samples and its complexity, the limitation of the experiment in using only four datasets is insufficient to draw a conclusion because of varied performances obtained in the experiment.

REFERENCE:

Akbani, R., Kwek, S., & Japkowicz, N. (2004). Applying Support Vector Machines to Imbalanced Datasets. *Machine Learning: ECML 2004*, 39-50. doi: 10.1007/978-3-540-30115-8_7

Beyan, C., & Fisher, R. (2018). Classifying imbalanced data sets using similarity based hierarchical decomposition. Retrieved from http://dx.doi.org/10.1016/j.patcog.2014.10.032

Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2009). Safe-Level-SMOTE: Safe-Level-Synthetic Minority Over-Sampling TEchnique for Handling the Class Imbalanced Problem. *Advances In Knowledge Discovery And Data Mining*, 475-482. doi: 10.1007/978-3-642-01307-2_43

Cateni, S., Colla, V., & Vannucci, M. (2018). A method for resampling imbalanced datasets in binary classification tasks for real-world problems. Retrieved from http://dx.doi.org/10.1016/j.neucom.2013.05.059

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, W. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal Of Artificial Intelligence Research*, *16*, 321-357. doi: 10.1613/jair.953

Cieslak, D., Chawla, N., & Striegel, A. Combating imbalance in network intrusion datasets. *2006 IEEE International Conference On Granular Computing*. doi: 10.1109/grc.2006.1635905

Douzas, G., Bacao, F., & Last, F. (2018). Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *Information Sciences*, *465*, 1-20. doi: 10.1016/j.ins.2018.06.056

Fernandez, A., Garcia, S., Herrera, F., & Chawla, N. (2018). SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. *Journal Of Artificial Intelligence Research*, *61*, 863-905. doi: 10.1613/jair.1.11192

Garcia, E. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE International Joint Conference On Neural Networks (IEEE World Congress On Computational Intelligence)*. doi: 10.1109/ijcnn.2008.4633969

Gonzalez-Abril, L., Angulo, C., Nuñez, H., & Leal, Y. (2017). Handling binary classification problems with a priority class by using Support Vector Machines. Applied Soft Computing, 61, 661-669. doi: 10.1016/j.asoc.2017.08.023

Hao, M., Wang, Y., & Bryant, S. (2014). An efficient algorithm coupled with synthetic minority over-sampling technique to classify imbalanced PubChem BioAssay data. Analytica Chimica Acta, 806, 117-127. doi: 10.1016/j.aca.2013.10.050

Han, H., Wang, W., & Mao, B. (2005). Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. *Lecture Notes In Computer Science*, 878-887. doi: 10.1007/11538059_91

Hartono, H., Sitompul, O., Tulus, T., & Nababan, E. (2018). Biased support vector machine and weighted-smote in handling class imbalance problem. *International Journal Of Advances In Intelligent Informatics*, *4*(1), 21. doi: 10.26555/ijain.v4i1.146

Hu, S., Liang, Y., Ma, L., & He, Y. (2009). MSMOTE: Improving Classification Performance When Training Data is Imbalanced. *2009 Second International Workshop On Computer Science And Engineering*. doi: 10.1109/wcse.2009.756

Jian, C., Gao, J., & Ao, Y. (2018). A new sampling method for classifying imbalanced data based on support vector machine ensemble. Retrieved from http://dx.doi.org/10.1016/j.neucom.2016.02.006

Khozeimeh, F., Alizadehsani, R., Roshanzamir, M., Khosravi, A., Layegh, P., & Nahavandi, S. (2018). An expert system for selecting wart treatment method. Retrieved from http://dx.doi.org/10.1016/j.compbiomed.2017.01.001

Kavakiotis, I., Tsave, O., Salifoglou, A., Maglaveras, N., Vlahavas, I., & Chouvarda, I. (2018). Machine Learning and Data Mining Methods in Diabetes Research. Retrieved from http://dx.doi.org/10.1016/j.csbj.2016.12.005

Loyola-González, O., Martínez-Trinidad, J., Carrasco-Ochoa, J., & García-Borroto, M. (2018). Study of the impact of resampling methods for contrast pattern based classifiers in imbalanced databases. Retrieved from http://dx.doi.org/10.1016/j.neucom.2015.04.120

López, V., Fernández, A., García, S., Palade, V., & Herrera, F. (2018). An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Retrieved from http://dx.doi.org/10.1016/j.ins.2013.07.007

Mi, Y. (2013). Imbalanced Classification Based on Active Learning SMOTE. *Research Journal Of Applied Sciences, Engineering And Technology*, *5*(3), 944-949. doi: 10.19026/rjaset.5.5044

Nekooeimehr, I., & Lai-Yuen, S. (2018). Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets. Retrieved from

Piri, S., Delen, D., & Liu, T. (2018). A synthetic informative minority over-sampling (SIMO) algorithm leveraging support vector machine to enhance learning from imbalanced datasets. *Decision Support Systems*, *106*, 15-29. doi: 10.1016/j.dss.2017.11.006

Prusty, M., Jayanthi, T., & Velusamy, K. (2017). Weighted-SMOTE: A modification to SMOTE for event classification in sodium cooled fast reactors. *Progress In Nuclear Energy*, *100*, 355-364. doi: 10.1016/j.pnucene.2017.07.015

Peng, Y., (2015). Adaptive sampling with optimal cost for class-imbalance learning. Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. p.2921-2927, Austin, Texas

Rajesh, K., & Dhuli, R. (2018). Classification of imbalanced ECG beats using re-sampling techniques and AdaBoost ensemble classifier. Biomedical Signal Processing And Control, 41, 242-254. doi: 10.1016/j.bspc.2017.12.004

Raudys, S., & Jain, A. (1991). Small sample size effects in statistical pattern recognition: recommendations for practitioners. IEEE Transactions On Pattern Analysis And Machine Intelligence, 13(3), 252-264. doi: 10.1109/34.75512

Schrider, D., & Kern, A. (2018). Supervised Machine Learning for Population Genetics: A New Paradigm. *Trends In Genetics*, *34*(4), 301-312. doi: 10.1016/j.tig.2017.12.005

Sharma, S., & Sharma, V. (2016). Performance of Various Machine Learning Classifiers on Small Datasets with Varying Dimensionalities: A Study. Circulation In Computer Science, 1(1), 30-35. doi: 10.22632/ccs-2016-251-23

Zhu, B., Baesens, B., & vanden Broucke, S. (2018). An empirical comparison of techniques for the class imbalance problem in churn prediction. Retrieved from http://dx.doi.org/10.1016/j.ins.2017.04.015

APPENDIX

Sources of Datasets

Dataset Biomarker:
https://www.researchgate.net/publication/322832878_Dataset_biomarkers

Dataset Hepatitis: https://archive.ics.uci.edu/ml/datasets/Hepatitis

Dataset Immunotherapy: https://archive.ics.uci.edu/ml/datasets/Immunotherapy+Dataset

Dataset Echocardiogram: https://archive.ics.uci.edu/ml/datasets/Echocardiogram

Code written in python for SIMO algorithm,

```
# importing the packages and modules
from sklearn import svm
from collections import Counter
from sklearn.metrics import confusion_matrix
import Euclidean
from sklearn.model_selection import train_test_split
from imblearn.over_sampling import SMOTE
from math import ceil
import numpy as np
import pandas as pd
from math import sqrt
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc


#defining the function SIMO
def SIMO(X,y, delta = 0.8, p = 0.5):      # delta and p value varies with the data
    X_train, X_val, y_train, y_val = train_test_split(X, y, stratify = y, test_size = 0.20)
    c = Counter(y)
    minority_class = 0 if c[0] < c[1] else 1
```

```
    minority_indices = []
    for ix, item in enumerate(y_train):
        if item == minority_class:
            minority_indices.append(ix) #finding minority data points index in training data
    Minority_Count= len(minority_indices)
    Majority_Count= len(y_train) - Minority_Count
    Imbalanced_gap=Majority_Count-Minority_Count

# training SVM to obtain informative data instances
    clf = svm.SVC(kernel='linear')
    clf.fit(X_train,y_train)
    predicted1 = clf.predict(X_val)
    w = clf.coef_
    b = clf.intercept_

# initialization
    CM_initial = confusion_matrix(y_val,predicted1)
    TN_initial = CM_initial[0][0]
    FN_initial = CM_initial[1][0]
    TP_initial = CM_initial[1][1]
    FP_initial = CM_initial[0][1]
    TPR = TP_initial/(TP_initial+FN_initial)
    TNR = TN_initial/(TN_initial+FP_initial)
    x = TPR * TNR
    G_Mean_initial = sqrt(x)

    G_Mean=G_Mean_initial
    Generated_Data_Count=0
    SVMModel=clf
    Predictors= np.array(X_train)
    Target=np.array(y_train)
```

```
    G_Mean_variation=G_Mean_initial
    Generated_Data=Imbalanced_gap
#creating a list
    X_list = []
    y_list = []
    g_mean_list = []


# loop terminates once generated synthetic instances equals imbalanced gap
    while  Generated_Data_Count < Imbalanced_gap:
        ind = [ix for ix, item in enumerate(Target) if item == minority_class]  #finding minority data
points index
        majority_ind = [ix for ix, item in enumerate(Target) if item != minority_class]
        #import pdb; pdb.set_trace()
        Mino = Predictors[ind]  #minority data points
        Mino_Target=Target[ind]
        majority_predictors = Predictors[majority_ind]
        majority_targets = Target[majority_ind]


#defining and calculating the minority data point distance from the decision boundary
        dis = []
        for i in range(Mino.shape[0]):
            #import pdb; pdb.set_trace()
            dis.append(Euclidean.EucledianDistance(Mino[i],w,b))
            #import pdb; pdb.set_trace()


 #finding informative minority data points
        dis_sort = np.sort(dis)
        dis_top_delta = dis_sort[0: int(ceil(delta * Mino.shape[0]))]
        # dis_top_delta=dis_sort[ceil(delta*size(Mino))]
        inf_p_ind=[]    #informative minority data points
        non_inf_p_ind = []
        for index, distance in enumerate(dis):
```

```
            if distance in dis_top_delta:
                inf_p_ind.append(index)
            else:
                non_inf_p_ind.append(index)
        inf_predictors = Mino[inf_p_ind]
        inf_target = Mino_Target[inf_p_ind]
        non_inf_predictors = Mino[non_inf_p_ind]
        non_inf_target = Mino_Target[non_inf_p_ind]
        new_predictors = np.concatenate([inf_predictors, majority_predictors], axis = 0)
        new_target_list = list(inf_target)
        new_target_list.extend(list(majority_targets))
        new_target = np.array(new_target_list)

# callable function is written to decide on ‘p’
        def _handle_p(y):
            class_counter = Counter(y)
            minority_class = 0 if class_counter[0] < class_counter[1] else 1
            minority_count = class_counter[minority_class]
            class_counter[minority_class] += int(ceil(minority_count * p))
            #import pdb; pdb.set_trace()
            return class_counter

# Over-sampling the informative minority data points using SMOTE
        sm = SMOTE(kind='regular', ratio = _handle_p, )
        Mino_Xres1, Mino_yres1 = sm.fit_sample(new_predictors, new_target)
        #import pdb; pdb.set_trace()
        Generated_Data_Count += len(Mino_Xres1) - len(new_predictors)
        Predictors =  np.concatenate([non_inf_predictors, Mino_Xres1], axis = 0)
        new_target_2_list = list(Mino_yres1)
        new_target_2_list.extend(list(non_inf_target))
        new_target_2 = np.array(new_target_2_list)
```

```
        Target = new_target_2

        # train SVM on resampled data
        clf = svm.SVC(kernel='linear')
        clf.fit(Predictors,Target)
        predicted2 = clf.predict(X_val)
        w = clf.coef_
        b = clf.intercept_

        CM = confusion_matrix(y_val,predicted2)
        TN = CM[0][0]
        FN = CM[1][0]
        TP = CM[1][1]
        FP = CM[0][1]
        TPR = TP/(TP+FN)
        TNR = TN/(TN+FP)
        x = TPR * TNR
        G_Mean = sqrt(x)
        G_Mean_variation=G_Mean
        g_mean_list.append(G_Mean)
        X_list.append(Predictors)
        y_list.append(Target)

# to find sampled data which fetched maximum g_mean on training SVM
    max_g_mean_index = g_mean_list.index(max(g_mean_list))
    final_resampled_X = X_list[max_g_mean_index]
    final_resampled_y = y_list[max_g_mean_index]

    #final_resampled_X = final_resampled_X.to_records(index=False)
    #final_resampled_X = np.array(final_resampled_X.tolist())
    Generated_Data=Generated_Data_Count
```

```
    return final_resampled_X, final_resampled_y
```

Code for Euclidean function used in SIMO

```
import numpy as np
def EucledianDistance(x, w, b):
    w_square= []
    for i in w:
        w_square.append(i**2)
    w_square
    k = (np.dot(w, np.transpose(x)))+b
    t = np.sqrt(np.sum(w_square))
    euc_d = abs(k)/t
    return euc_d
```

Code for Modelling

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from imblearn.over_sampling import SMOTE
from imblearn.over_sampling import ADASYN
from sklearn.linear_model import LogisticRegression
from sklearn import model_selection
from sklearn import metrics
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
import matplotlib.pyplot as plt
from sklearn.svm import SVC
import SIMO_B
```

```
from collections import Counter
from sklearn.model_selection import GridSearchCV
from scipy import stats


data = pd.read_csv('C:\\Document\\D17124483\\Dissertation\\Imbalanced dataset\\echocardiogram.csv', sep = ',') # changes for each dataset
data.shape # to find the dimensions


# fetching unique values of each variable to check if there are any irrelevant figures
[data[col_name].unique() for col_name in data.columns]


# replace if any unique values are found in the previous step
data = data.replace(2, np.nan)  #data dependent
data = data.replace('', np.nan)
data = data.replace('NA', np.nan)
data = data.replace(77, np.nan)  #data dependent


# common code for all the datasets in fetching missing value report, descriptive statistics report and to build predictive models using Logistic regression and SVM
class Analysis():
  def exp_analysis(X):
      descriptive_analysis = X.describe()
      print('\n\n Descriptive analysis \n\n',descriptive_analysis)
      print(X.shape)      #exploratory data analysis


   def missing_values(X):
      print ("\n\nMissing value analysis:")
      quan = list(X.loc[:,X.dtypes != 'object'].columns.values )
      qual = list(X.loc[:,X.dtypes == 'object'].columns.values)
```

```
        print("quan:\n\n", quan)
        print("\n\nqual:\n\n", qual)
        print ("\n\n Numerical variables:")
        total_quan = X[quan].isnull().sum().sort_values(ascending=False)
        percent_quan = (X[quan].isnull().sum() /
X[quan].isnull().count()).sort_values(ascending=False)
        missing_data_quan = pd.concat([total_quan, percent_quan], axis=1, keys =['Missing',
'Percent'])
        print(missing_data_quan)
        print("**"*40)
        print ("\n\n Categorical variables:")
        total_qual = X[qual].isnull().sum().sort_values(ascending=False)
        percent_qual = (X[qual].isnull().sum() /
X[qual].isnull().count()).sort_values(ascending=False)
        missing_data_qual = pd.concat([total_qual, percent_qual], axis=1, keys =['Missing',
'Percent'])
        print(missing_data_qual)  # missing value analysis


# Logistic regression
    def predictive_models_LR(X_train, y_train, X_test, y_test):
        param_grid = [{'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000]}, {'penalty' : ['l1','l2']}]
        logreg = GridSearchCV(LogisticRegression(), param_grid, cv = 5)
        logreg.fit(X_train, y_train)
        predict = logreg.predict(X_test)
        accuracy = metrics.accuracy_score(y_test, predict)
        print('test accuracy:', accuracy)
        print(confusion_matrix(y_test, predict))
        print(classification_report(y_test, predict))
        CM = confusion_matrix(y_test,predict)
        TN = CM[0][0]
```

```
FN = CM[1][0]
TP = CM[1][1]
FP = CM[0][1]
G_Mean = ((TP/(TP+FN))*(TN/(TN+FP)))**(1/2)
print('G_Mean', G_Mean)
print('G_Mean', G_Mean)
print("TP", TP)
print("FP", FP)
print("FN", FN)
print("TN", TN)
FPR = FP/(FP+TN)
print("FPR", FPR)

 # View best hyperparameters
print('Best Penalty:', logreg.best_estimator_.get_params()['penalty'])
print('Best C:', logreg.best_estimator_.get_params()['C'])

# calculate the fpr and tpr for all thresholds of the classification
fpr, tpr, threshold = metrics.roc_curve(y_test, predict)
roc_auc = metrics.auc(fpr, tpr)

# ROC plot
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
```

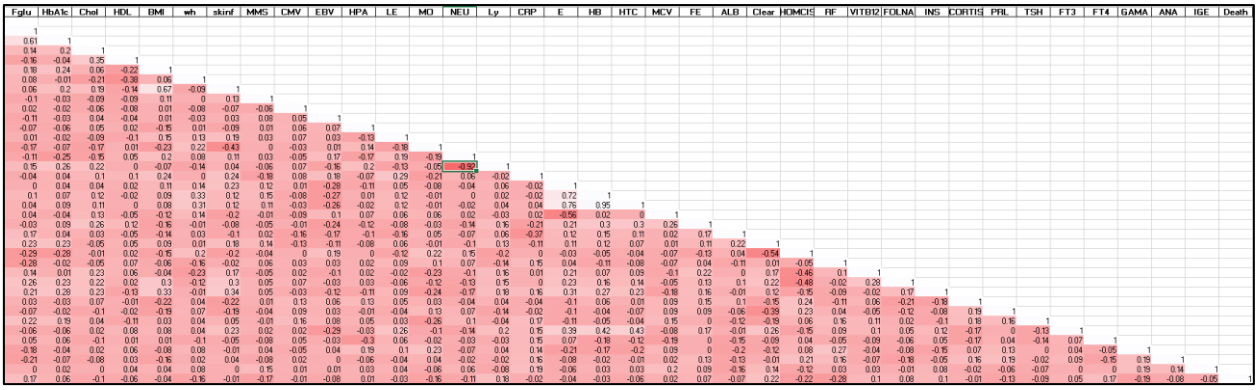
```
        plt.ylabel('True Positive Rate')
        plt.xlabel('False Positive Rate')
        plt.show()

# SVM
    def predictive_models_SVM(X_train, y_train, X_test, y_test):
        param_grid = [{'kernel': ['linear']},{'C': [1, 2, 3, 4, 5, 6,7, 8, 9, 10]},
                      {'gamma':[0.01, 0.02, 0.03, 0.04, 0.05, 0.10, 0.2, 0.3,0.4, 0.5]}]
        SVM_linear = GridSearchCV(SVC(), param_grid, cv =5)
        SVM_linear.fit(X_train, y_train)
        predict = SVM_linear.predict(X_test)
        accuracy = metrics.accuracy_score(y_test, predict)
        print('test accuracy:',metrics.accuracy_score(y_test, predict))
        print(confusion_matrix(y_test,predict))
        print(classification_report(y_test,predict))
        CM = confusion_matrix(y_test,predict)
        TN = CM[0][0]
        FN = CM[1][0]
        TP = CM[1][1]
        FP = CM[0][1]
        G_Mean = ((TP/(TP+FN))*(TN/(TN+FP)))**(1/2)
        print('G_Mean', G_Mean)
        print("TP", TP)
        print("FP", FP)
        print("FN", FN)
        print("TN", TN)
        FPR = FP/(FP+TN)
        print("FPR", FPR)
```

```
        # View best hyperparameters
        print('Best Gamma:', SVM_linear.best_estimator_.get_params()['gamma'])
        print('Best C:', SVM_linear.best_estimator_.get_params()['C'])

        # calculate the fpr and tpr for all thresholds of the classification
        fpr, tpr, threshold = metrics.roc_curve(y_test, predict)
        roc_auc = metrics.auc(fpr, tpr)

        # ROC plot
        plt.title('Receiver Operating Characteristic')
        plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
        plt.legend(loc = 'lower right')
        plt.plot([0, 1], [0, 1],'r--')
        plt.xlim([0, 1])
        plt.ylim([0, 1])
        plt.ylabel('True Positive Rate')
        plt.xlabel('False Positive Rate')
        plt.show()

# calling the class to retrieve necessary information
Analysis.missing_values(data)

# Imputation- dataset dependent
data.alive.fillna(data.alive.mode()[0], inplace=True)
data.epss.fillna(data.epss.median(),inplace=True)
data.lvdd.fillna(data.lvdd.median(),  inplace=True)
data.pericardialeffusion.fillna(data.pericardialeffusion.mode()[0],  inplace=True)
```

```
data.fractionalshortening.fillna(data.fractionalshortening.median(),  inplace=True)
data.age.fillna(data.age.median(),  inplace=True)
data.w_score.fillna(data.w_score.median(),  inplace=True)
data.w_index.fillna(data.w_index.median(), inplace=True)
data.survival.fillna(data.survival.median(),  inplace=True)

#calling after imputation to fetch descriptive statistics
Analysis. exp_analysis(data)

# segregate independent and target variables into X and y
X= data.drop('alive', axis=1)  # target is data dependent
y = data['alive']

# data partition
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.25)

#Normalising the data
quan = list(X_train.columns[X_train.dtypes != 'object'])
scaler = StandardScaler()
X_train[quan] = scaler.fit_transform(X_train[quan])
X_test[quan] = scaler.transform(X_test[quan])

#one hot encoding
print("\n\nOne hot Encoding.....\n")
label_encoder = LabelEncoder()
y_train = label_encoder.fit_transform(y_train)
y_test = label_encoder.transform(y_test)
qual = list(X_train.loc[:,X_train.dtypes == 'object'].columns.values)
```

```
X_cat1 = pd.get_dummies(X_train[qual])
X_train = pd.concat([X_train, X_cat1], axis=1)
X_train = X_train.drop(qual,axis =1)
print(X_train)
print(y_train)

X_cat2 = pd.get_dummies(X_test[qual])
X_test = pd.concat([X_test, X_cat2], axis=1)
X_test = X_test.drop(qual,axis =1)
print(X_test)
print(y_test)

# building Models for imbalanced data set
print("Imbalanced_Linear.....")
Analysis.predictive_models_LR(X_train, y_train, X_test, y_test)
print("Imbalanced_SVM.....")
Analysis.predictive_models_SVM(X_train, y_train, X_test, y_test)

 #SMOTE oversampling
sm1 = SMOTE(kind = 'regular', ratio = 'minority' )
X_res1, y_res1 = sm1.fit_sample(X_train, y_train)
sm2 = SMOTE(kind = 'borderline1', ratio = 'minority' )
X_res2, y_res2 = sm2.fit_sample(X_train, y_train)
sm3 = SMOTE(kind = 'borderline2', ratio = 'minority' )
X_res3, y_res3 = sm3.fit_sample(X_train, y_train)

# building models on SMOTE resampled data
print("SMOTE-regular-LR.....")
```

```
Analysis.predictive_models_LR(X_res1, y_res1, X_test, y_test)
print("SMOTE-borderline1-LR.....")
Analysis.predictive_models_LR(X_res2, y_res2, X_test, y_test)
print("SMOTE-borderline2-LR.....")
Analysis.predictive_models_LR(X_res3, y_res3, X_test, y_test)

print("SMOTE-regular-SVM.....")
Analysis.predictive_models_SVM(X_res1, y_res1, X_test, y_test)
print("SMOTE-borderline1-SVM.....")
Analysis.predictive_models_SVM(X_res2, y_res2, X_test, y_test)
print("SMOTE-borderline2-SVM.....")
Analysis.predictive_models_SVM(X_res3, y_res3, X_test, y_test)

#ADASYN
print("ADASYN-LR.....")
ada = ADASYN()
X_resampled, y_resampled = ada.fit_sample(X_train, y_train)
Analysis.predictive_models_LR(X_resampled, y_resampled, X_test, y_test)
print("ADASYN-SVM.....")
Analysis.predictive_models_SVM(X_resampled, y_resampled, X_test, y_test)

#SIMO
print("SIMO-LR.....")
X_simo, y_simo = SIMO_B.SIMO(X_train, y_train)
Analysis.predictive_models_LR(X_simo, y_simo, X_test, y_test)
print("SIMO-SVM.....")
Analysis.predictive_models_SVM(X_simo, y_simo, X_test, y_test)
```



Correlation matrix for dataset ‘Biomarker’