Handling Class Imbalance Problem using Oversampling and Ensemble Learning Techniques

Dissertation

Submitted in partial fulfillment of the requirement for the degree of

Master of Technology

In

Computer Science and Engineering

Under the Supervision of

Prof. Anjana Gosain

USICT, GGSIPU

Submitted By:

Neha Gahlan

(Roll No.: 0176404819)

University School Of Information, Communication and Technology

Guru Gobind Singh Indraprastha University

New Delhi- 110078

July 2021

DECLARATION

This is to certify that Thesis entitled " Handling Class Imbalance Problem using Oversampling and Ensemble Learning Techniques" which is submitted by me in partial fulfillment of the requirement for the award of degree M.Tech. in Computer Science Engineering to USICT, GGSIP University comprises only my original work and due acknowledgement has been made in the text to all other material used.

Date: 30 June 2021

APPROVED BY ....
Dean
USICT, GGS Indraprastha University
Sector -16C, Dwarka, New Delhi

CERTIFICATE

This is to certify that the Ms. Neha Gahlan (Roll no. 01716404819) has prepared this dissertation with exhaustive study of the subject. The thesis prepared by Ms. Neha Gahlan is innovative and her own idea to solve the objective of the given area.

Date: 30 June 2021

Professor Anjana Gosain
USICT, GGS Indraprastha University
Sector -16C, Dwarka, New Delhi

ACKNOWLEDGEMENT

It gives me great pleasure to express my deep sense of gratitude and indebtedness to my mentor Prof. Anjana Gosain for her valuable support and encouraging mentality throughout the thesis. I am highly obliged to her for providing me this opportunity to carry out the ideas and work during my thesis period and helping me to gain the successful completion of my Minor Thesis.

I am highly grateful to Prof. Pravin Chandra (Dean of School Of Information, Communication and Technology) for giving me the proper guidance and advice and facility for the successful completion of my Minor Thesis.

Date: 30 June 2021

Neha Gahlan
M.tech CSE
01716404819

LIST OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

In the realm of artificial intelligence, there exists a skewed distribution of classes, which contains an imbalanced ratio of instances. The majority class does have a large no. of instances, whereas the minority class has less no. of instances. The presence of such a scenario in data is called Class Imbalance Problem (CIP). Due to the presence of CIP in the data, the resulting models perform less than optimal. The presence of majority instances misleads the accuracy and other performance measures of the classifier model. The focus of this thesis will be on the two approaches to addressing the problem of class imbalance. SMOTE - an oversampling technique used in the first approach, whereas the stacking ensemble algorithm used in the second. The results reveal that the performed approaches provided the highest values of performance metrics such as roc score, accuracy, geometric mean, and f-measure when compared to other classification models on five datasets sourced from the UCI machine learning repository with different unbalanced class ratios. The proposed approach in this thesis is to use stacking between Naïve Bayes and SVM algorithms along with the oversampling technique SMOTE.

Keywords*- CIP, Classification, SMOTE, Stacking.*

CHAPTER 1

INTRODUCTION

The technique of looking at vast databases of data in order to generate new data is known as data mining. Data mining entails extracting new data, but this isn't the case; rather, data mining entails extrapolating patterns and new information from data you've already gathered. Dependency detection, class identification, class description, and outlier/exception identification are all parts of data mining, with the latter focusing on a small fraction of data points that are generally dismissed as noise. Outliers have been considered by some machine learning and data mining algorithms, but only to the extent of accepting them in whatever the algorithms are supposed to do. The precise definition of an outlier is frequently based on underlying assumptions about the data structure and the detection technology used [31].

Data mining includes analysis and prediction. Statistical models, machine learning approaches, and mathematical algorithms like neural networks and decision trees can all be used in these technologies.

Some of the data mining approaches are categorised, as follows:

- Classification: This method is used to extract vital and relevant data and metadata. This data mining technique aids in the classification of data into several categories.

- Clustering: Clustering is the partitioning of data into groups of related items. The data is described by a few clusters, which sacrifices some details but improves the overall quality. It uses clusters to model data.

- Regression: Regression analysis is a data mining technique for determining and analysing the connection between variables when another component is present. It's used to figure out how likely something is.

- Association Rules: This data mining method aids in the discovery of a connection between two or more things. In the data set, it uncovers a hidden pattern.

- Sequential Patterns: It entails identifying interesting subsequences within a set of sequences, with the value of a sequence being quantified using several parameters such as length, occurrence frequency, and so on.

- Outlier Detection: In many circumstances, simply finding an overall pattern will not provide complete picture of your data set. Anomalies, or outliers, in your data must also be identified.

In the context of mining techniques, classification is crucial. This is a procedure for classifying data, as the name implies. In addition, several decisions must be made in order to bring the data together. It frequently hinges on a set of input variables. A succession of acknowledgements and data inputs are used to determine the classification.

Classification is a predictive modelling task in machine learning where a class label is predicted for a given example of input data. The training dataset will be used to calculate the optimum way to map samples of input data to specified class labels. As a result, the training dataset needs to be sufficiently representative of the problem and contain a large number of samples of each class label.

These are the different types of tasks that classification encounters:

- Binary Classification
- Multi Class Classification
- Multi-Label Classification
- Imbalanced Classification

The algorithms used for classification are: Respective to Machine Learning

- Logistic Regression
- K-Nearest Neighbors
- Decision Trees
- Support vector machine
- Naïve Bayes
- Random Forest
- Gradient Boosting

Class Imbalance Problem

The imbalanced distribution of classes in datasets appears when the proportion of one class has a higher ratio than the other class. Class having large number of instances is called majority class and the ones having less number of instances is called minority class. The underrepresented classes i.e. the minority classes are apparently anticipated as rare events, or presumed as noise or outliers, which lead to more misclassification of minority classes [17]. Oil spill detection, credit card frauds, shuttle system failure, sentiment analysis, web spam detection, risk management and nuclear explosion, video mining, text mining, medical and fault diagnosis, anomaly detection, and so on are examples of situations where the minority class is of greater interest than the majority class. The minority class is more concerned with and important than the dominant class. The Class Imbalance Problem occurs when typical classification methods fail to accurately categorise the minority class. The problem of class imbalance has a considerable impact on performance and presents significant hurdles for machine learning approaches. In recent years, the class imbalance problem has been discovered in many practical domains and has become a hot topic in machine learning. Almost all of the instances in such a problem are classified as one class, while much fewer are labelled as the other, usually the more important class.

The focus is on detecting data points in the minority group, and that can present some common problems. Typically in situations like this the data you have collected is imbalanced, meaning that the target you are interested in has significantly smaller amounts of data than the other groups [15]. This imbalance in your data will cause your model to become bias towards selecting the majority group [17]. Also this unbalanced data might provide good classification accuracy, but has detrimental impact on classification performance measures such as ROC score, g-mean, f-measure and so on. Researchers have proposed a number of ways to handle the imbalance class problem, which are divided into three into these categories:

- Data level,
- Algorithmic level, and
- Hybrid form.

Data level - Datasets are balanced first, then traditional classification methods are employed at the data level approach. There are two methods in data level approach. These are the two techniques for rebalancing the datasets:

- Undersampling is eliminating instances of the majority class - There have been multiple heuristic under-sampling methods proposed or introduced from data cleaning in recent years. They are based on either of two different noise model hypotheses. One thinks examples that are near to the classification boundary of the two classes are noise, while the other considers examples with more neighbors of different labels are noise. The most naive undersampling method is random under-sampling, a non-heuristic method trying to balance class distributions through the random elimination of majority class examples. This leads to discarding potentially useful data that could be important for classifiers. Some of these are mentioned as below:

    a. Random Undersampling (RUS)
    b. Near Miss
    c. Condensed Nearest Neighbor Rule

- Oversampling is adding new minority instances to the datasets - Random over sampling is a non-heuristic method that aims to balance Class distributions through the random replication of minority class examples. Random over-sampling has two shortcomings. First, it will increase the likelihood of occurring over-fitting, since it makes exact copies of the minority class examples. Second, oversampling makes learning process more time consuming if the original data set is already fairly large but imbalanced. Some of these are mentioned as below:

    a. Random Oversampling
    b. SMOTE
    c. ADASYN

Algorithm level - new classification algorithms are built that improve current ones without modifying the original dataset. To lower the overall cost, the cost sensitive method assigns each class a different weight. When training the machine-learning model, it considers the cost of prediction errors. The Ensemble method combines several classifiers to obtain higher accuracy, stability, and robustness. There are two types of methods in this approach:

- Ensemble methods - These are meta-learning algos that combine numerous machine learning techniques into a single predictive model to reduce variance (bagging), bias (boosting), or increase prediction accuracy (stacking). The approaches used in ensembles can be classified into two categories:

a. Sequential Ensemble - Base learners are generated successively in sequential ensemble methods. The primary motivation for sequential techniques is to take advantage of the base learners' interdependence. By giving previously mislabeled cases more weight, the total performance can be improved e.g. **AdaBoost.**

b. Parallel Ensemble - In parallel, basic learners are generated. As averaging may dramatically reduce error, the primary purpose of the parallel technique is to use independent base learners like in - **Stacking.**

- Cost sensitive algorithms – It is a sort of learning which considers the costs of misclassification. The purpose of this learning is to keep the total cost as low as possible. Misclassification costs are not taken into account in cost-insensitive learning.

a. Majority Class; Negative or no-event is assigned as class label 0.
b. Minority Class: A positive event or incident that has been labelled as a class 1.

Hybrid form - To come up with a better solution to the problem of imbalance class, a hybrid strategy integrates both the above-mentioned approaches - data level and algorithm level. Figure 1 shows conceptual framework of the hybrid machine learning workflow. Here multiple algorithms are combined in the initial model and then as a single model goes for training along with optimization. The predicted and target dataset are the inputs of the base classifier models. The results obtained by this hybrid method are much refined and better in comparison with single classifiers.

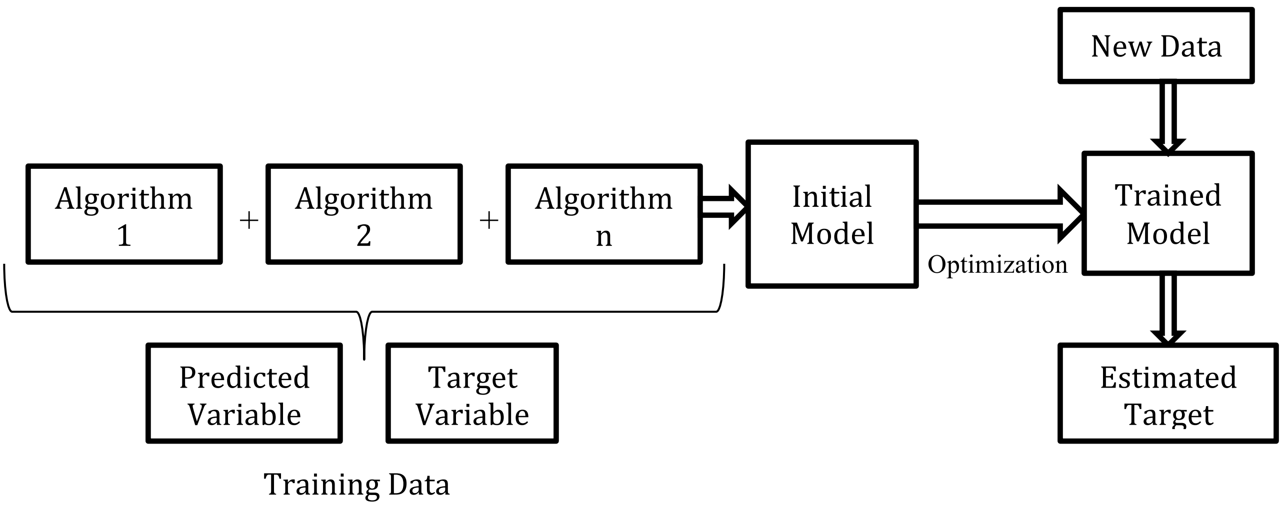


Figure 1. Hybrid workflow

We use hybrid method in this thesis: an oversampling technique – SMOTE (for the dataset), and an ensemble method called Stacking for the classifier level. We balanced the five datasets - Pima India Diabetes (PID), Breast Cancer Wisconsin, Statlog Heart, US Credit, and Spam Base - using the SMOTE method, and then used Stacking on the base classifiers SVM (Support Vector Machine), KNN (K-Nearest Neighbor) and Nave Bayes to get the performance measures - Accuracy, F-measure, G-mean, and Roc Auc. Based on the results, we discovered that stacking combined with Smote outperforms the other base classifiers.

The thesis is laid out as follows:

Chapter 2 describes The Literature Review including the work done by the researchers in the past while addressing the class imbalance problem arising during classification of the real world data. The proposed methods for handing class imbalance problem are mentioned. The researchers have worked on different Hybrid, Data-level and Algorithm level solutions such as Oversampling, Undersampling, Ensemble Methods, Cost Sensitive methods, etc. are reviewed in this section.

Chapter 3 describes The Proposed Approach in this thesis. We have used the ensemble method for the base learners and the oversampling technique for the dataset. The oversampling method used is SMOTE and the Ensemble method used is Stacking learning, for the classifiers SVM and Naïve Bayes as the base learner model.

The dataset analysis and results are represented in Chapter 4. In this thesis, we use five real-world datasets, sourced from the UCI machine learning repository with different unbalanced class ratios. The results for different evaluation measures are recorded- Accuracy, F-measure, G-mean and ROC AUC. The highest values are highlighted in table.

The thesis comes to a conclusion with Chapter 5. Based on the results, we discovered that stacking combined with Smote outperforms the other base classifiers. The result outcomes shows that Stacking algorithm along with SMOTE technique gives better performance than the other base models with or without smote, as not only the Accuracy score is improved but also F- Measure, G-mean and ROC AUC is also showing better results. So, we conclude that Class imbalance problem can be resolved by this proposed hybrid approach and can solve the imbalance distribution in the real world data.

CHAPTER 2

LITERATURE REVIEW

Several researches have been conducted on various techniques to coping with dataset class imbalances.

Hybrid Method - Literature Review:

Sajid Ahmed [4] in his study explained how to handle class imbalances in the dataset where, he used ensemble-resampling techniques like Rus-Bagging, Adasyn-Bagging, Smote-Bagging and Rysin-Bagging. His analysis revealed that the four techniques he used were successful in increasing the classification algorithm's performance.

Yingze Yang [5] in his research dubbed SMOTE-Boosting and employed ensemble with resampling smote. The results of his research revealed that proposed strategy might increase the classification algorithm's performance.

UlagaPriya and Pushpa [6] in their research, worked on the ensemble bagging and boosting along with smote and did the comparison among them and concluded that ensemble bagging and boosting with smote outperforms other ensemble techniques.

Cangzhi [7] in their research did a comparison on the performance of safe-level smote along with ensemble learning logistic regression and random forest along with knn and svm base classifiers. Oversampling is a method of improving the number of instances in a minority group.

G. E. A. P. A Batista [29]. When comparing the researched over-sampling methods, their findings demonstrate that Random over-sampling produced the smallest rise in the mean number of induced rules and Smote + ENN produced the smallest increase in the mean number of conditions per rule.

Oversampling - Literature Review

To address class imbalance problem, oversampling approaches such as Randshuff [12], SMOTE [9], SMOTE-Borderline [11], ADASYN [10] and SMOTE-Borderline [11] have been already proposed. They create synthetic data in order to expand the decision region. SMOTE is the oversampling technique used in this study along with Stacking Learning, which is an ensemble method that combines the classification algorithms Support Vector Machine and Nave Bayes.

Anjana and Saanchi [17] In their research, they tackled the issue of class imbalance by using Matlab to build four oversampling techniques: SMOTE, ADASYN, borderline-SMOTE, and Safe-level SMOTE. Their findings revealed that Safe Level SMOTE beats the other approaches in most datasets, with the highest f-measure and g-mean values.

Maheshwari, Aggarwal and Sharma [18]. To reduce the imbalance ratio, they over-sampled the minority class using mutation and crossover operators, and then utilised clustering for both classes to remove duplicate samples and noisy samples. As a result, the two methods are integrated and the samples of interest are preserved, enhancing computing efficiency.

J. A. Saez [22]. Their contribution proposes that SMOTE be enhanced with an Iterative-Partitioning Filter (IPF) noise filter to address these issues. In a controlled experimental examination against SMOTE and its most well-known generalisations, the features of this proposition are addressed. Their findings reveal that the novel proposal outperforms existing SMOTE generalisations in all of these scenarios.

H. He, Y. Bai [23]. A unique adaptive synthetic (ADASYN) sampling strategy for learning from imbalanced data sets is presented in this research. As a result, the ADASYN technique improves data distribution learning in two ways: (1) minimising the bias created by the class imbalance, and (2) adaptively pushing the classification decision boundary toward the difficult examples.

H. Han, W. Wang, and B. Mao [24]. This study introduces borderline-SMOTE1 and borderline-SMOTE2, two new minority over-sampling approaches that over-sample only the minority examples on the borderline. Their experiments show that their methods obtain higher TP rates and F-values for the minority class than SMOTE and random over-sampling strategies.

Ruchika and Shine [30] according to their research, when balanced datasets were employed using the oversampling technique SMOTE, the prediction error dropped and the effectiveness of machine learning algorithms improved

Ensemble - Literature Review

G. Zhou and F. Guo [32]. They demonstrated that diversity may be achieved by optimising the sampling strategy and giving a higher weight to the base classifier that can properly identify error-prone data samples, resulting in a larger margin value for this type of data in ensemble learning. As a result, the problem of maximising and minimising margin is reduced to adjusting the weights of base classifiers, and the diversity of Bagging is realised by scientifically and reasonably setting the weights vector of base classifiers to improve ensemble learning's generalisation ability and classification accuracy.

W. -Y. Lin and I. Dai [33]. They present an ensemble of ADR signal detectors that employs the AdaBoost ensemble learning approach. They suggested a system that combines the benefits of many ADR detection methods and adjusts the weight of each ADR detection approach automatically to improve overall detection effectiveness.

P. Kumkar, I. Madan, A. Kale, O. Khanvilkar and A. Khan [34]. Their research compares four ensemble methods for real estate appraisal: Bootstrap Aggregating, Random Forest, Gradient Boosting, and Extreme Gradient Boosting. The comparison is based on an estimate of Mumbai real estate prices. Web scraping from the real estate website 99acres provided the data for this investigation. The acquired data was used to train and test the above-mentioned ensemble models, as well as to compare their results. Grid search was utilised to fine-tune the learning models' hyperparameters.

C. Perales-González, F. Fernández-Navarro, M. Carbonero-Ruz and J. Pérez-Rodríguez [35]. The framework of NCL is examined in this article, and it is discovered that instead of reducing the residuals of the final ensemble, it minimises the combination of mistakes of the individual members of the ensemble. They offer a new ensemble framework called global negative correlation learning (GNCL), which focuses on global ensemble optimization rather than individual component fitness.

Y. Tian and X. Wang [36]. This work provides an SVM ensemble approach based on an improved Adaboost algorithm iteration process. In order to tackle the problem of Adaboost being susceptible to noise and having a long training period, the updated Adaboost algorithm includes methods for adding sample selection and feature selection in its iterative process.

Y. ZHANG, R. LU, J. HUANG and D. GAO [37]. To overcome the problem of class imbalance, an evolutionary-based ensemble under-sampling (EEU) approach is suggested in this study. In particular, an evolutionary method is utilised to under sample the data, and ensemble learning is employed to train several base classifiers. The advantage of this approach is that it can increase minority class accuracy. On five UCI datasets, comparison tests are conducted, and the findings show that EEU outperforms alternative sampling strategies.

Class Imbalance Problem - Literature Review
Nitesh, Kevin and Lawrence [19]. Their findings indicated that combining over-sampling and under-sampling can improve classifier performance when compared to merely under-sampling the majority class.

N.V. Chawla [20] Discuss some of the sampling approaches used to balance the datasets, as well as the performance metrics that are better suited to mining imbalanced datasets.

T. Jo and N. Japkowicz, [21] test an approach that considers the tiny disjunct problem and show that it produces results that are superior to those achieved using standard or advanced solutions to the class imbalance problem. Their experiments demonstrate that class imbalances do not directly cause the problem, rather class imbalances can result in tiny disjuncts, which can lead to degradation. They claim that, in order to improve classifier performance, focusing on the small disjuncts problem rather than the class imbalance problem may be more beneficial.

Guo, X., [26]. First, this thesis looked at academic activities that were specifically designed to address the issue of class disparity. Then, in four distinct levels, according to learning periods, He looked into several cures. This article finally showed some future directions after surveying evaluation metrics and several other associated elements.

G. E. A. P. A Batista [29]. When comparing the researched over-sampling methods, their findings demonstrate that Random over-sampling produced the smallest rise in the mean number of induced rules and Smote + ENN produced the smallest increase in the mean number of conditions per rule.

.

CHAPTER 3

PROPOSED APPROACH

Proposed Method

In this thesis, we have used the ensemble method for the base learners and the oversampling technique for the dataset. The oversampling method used is SMOTE and the Ensemble method used is Stacking learning, for the classifiers SVM and Naïve Bayes as the base learner model.

The dataset in this research is split into two parts: training and testing, each with 80% and 20% of the data. We used two scenarios for testing the models. In the first, original data is used for testing without oversampling and in the other oversampled data is used for testing.

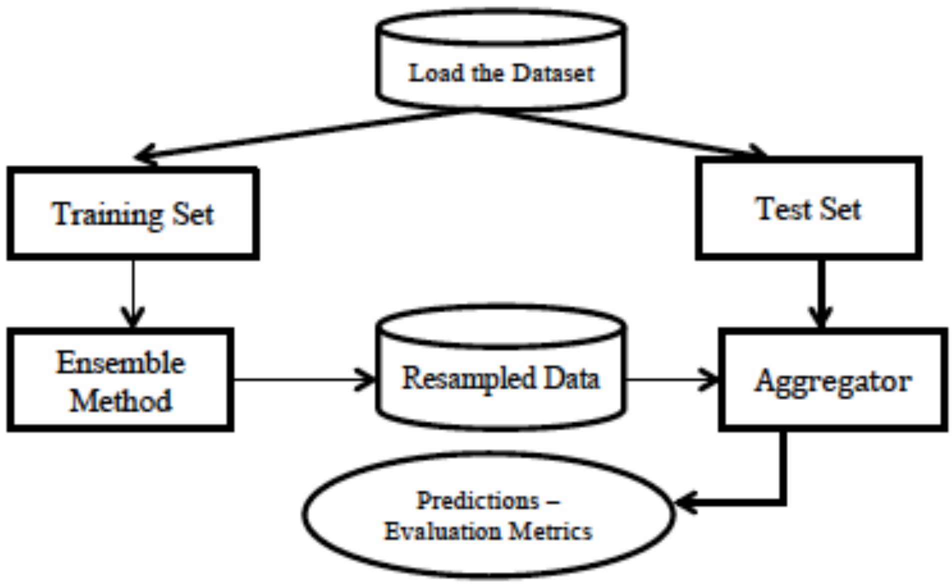


Figure 2. Proposed Approach

The workflow is explained in the below figure:

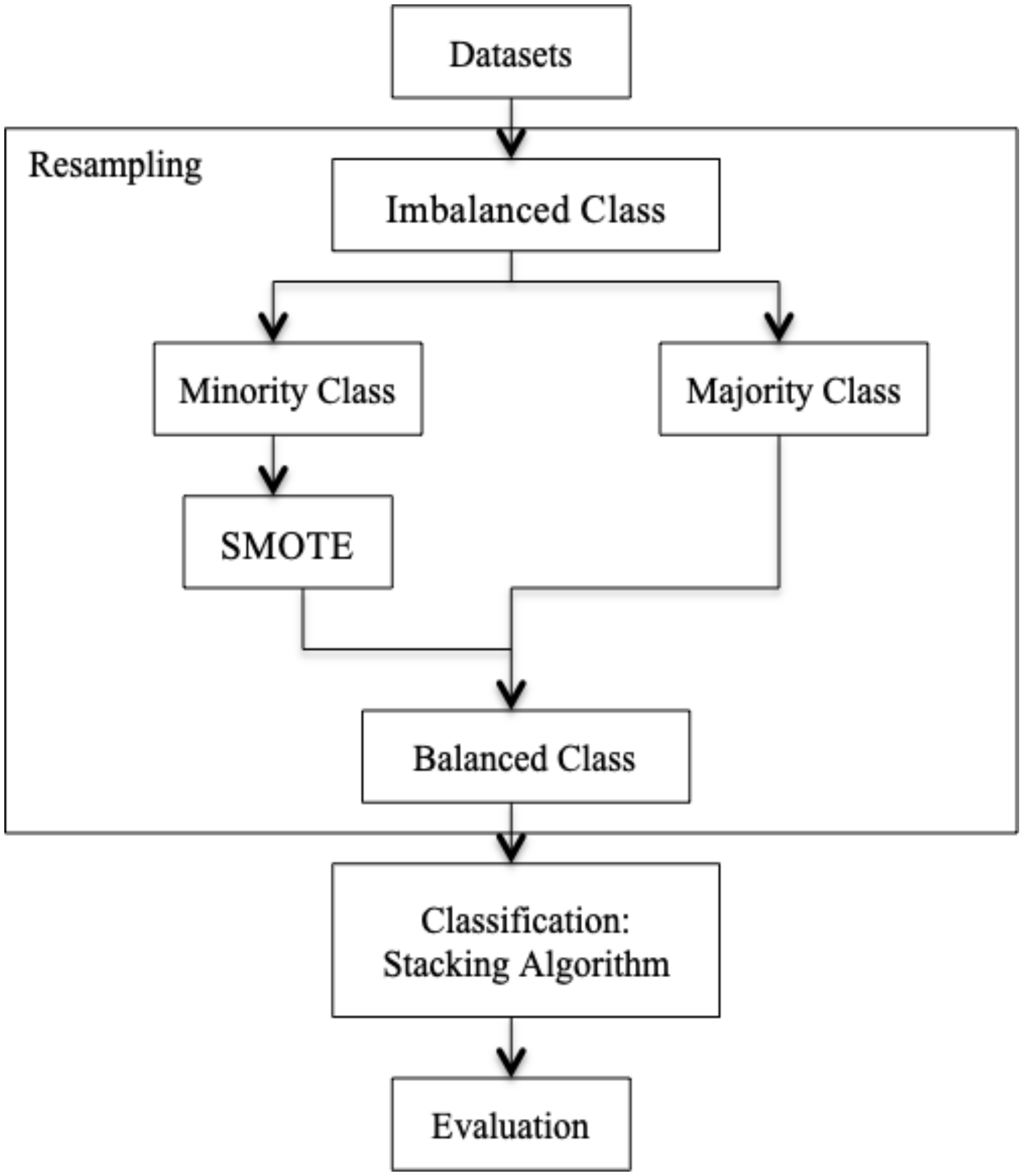


Figure 3. Research Workflow

1. Stacking

It is the ensemble method in which, data is split into training dataset, which is again split into further training and testing data. Different subsets of the training data are constructed, and each subset is evaluated to assess the performance of the classifier model. To establish a meta classifier, weights are assigned to each classifier in relative form [13]. The steps for stacking are explained as below:

- The initial training set is used to train the base classifiers.
- The classifiers generate predictions from the separate validation data.
- The meta-level training set includes the validation and prediction sets generated by the base classifiers in the validation set.
- The meta level training dataset is then used to train the final classifier or meta classifier.

Algorithm: Stacking

1. Input: training data $D = \{x_i, y_i\}^{m}_{i=1}$
2. Output: ensemble classifier *H*
3. *Step 1: learn base-level classifiers*
4. **for** *t = 1* to *T* **do**
5. learn h_t based on D
6. **end for**
7. *Step 2: construct new data set of predictions*
8. **for** *i* =1 to *m* **do**
9. $D_h = \{x'_i, y_i\}$, where $x'_i = \{h_1(x_i), \ldots, h_t(x_i)\}$
10. **end for**
11. *Step 3: learn a meta-classifier*
12. learn *H* based on D_h
13. return *H*

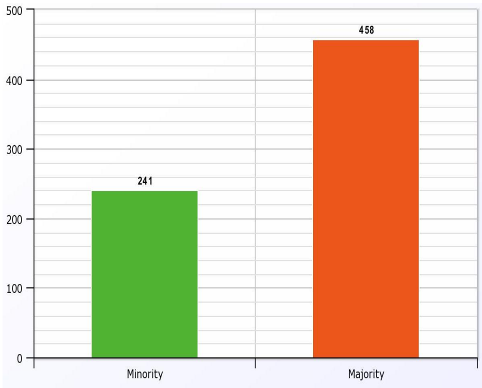
2. Resampling Using SMOTE

In this thesis, we use Safe level SMOTE. Before creating synthetic samples, it gives positive instances a safe level value [14]. The samples are generated synthetically in the minority class instead of replacing existing samples, in order to balance with the instances of the majority class. This method is used to create artificial minority cases. This SMOTE technique improves data distribution learning by decreasing bias induced by class imbalances, with the prime aim of balancing the minority class with fewer occurrences. Safe Level SMOTE, Regular SMOTE, Cluster SMOTE, Borderline SMOTE are some versions of SMOTE.

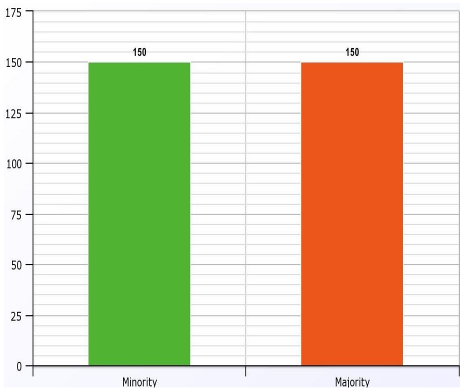
SMOTE technique is used to balance the dataset instances.

TABLE II. SMOTE application

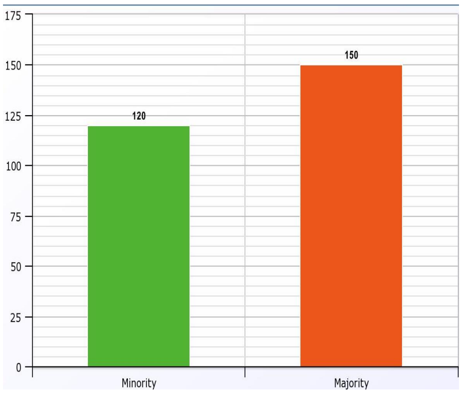
Datasets	**Before Smote**		**After Smote**	
	Minority instances	Majority instances	Minority instances	Majority instances
D1	268	500	500	500
D2	241	458	458	458
D3	120	150	150	150
D4	1813	2788	2788	2788
D5	300	700	700	700



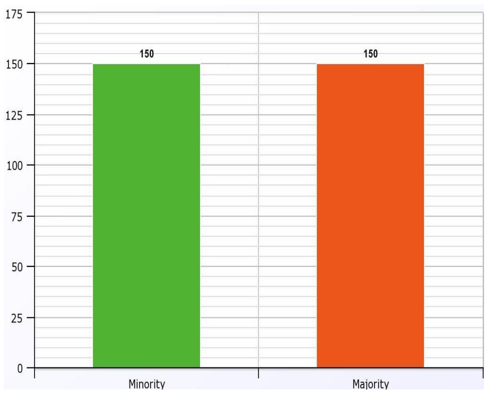
D1. Original



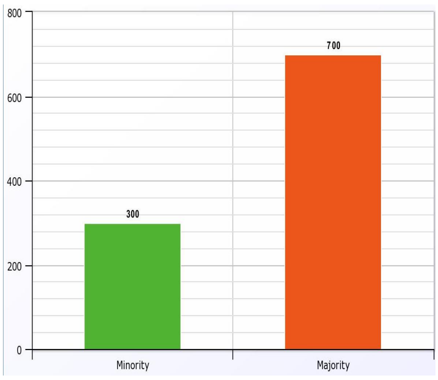
D1. Resampled



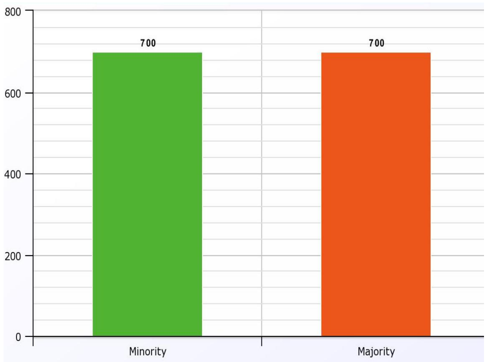
D2. Original



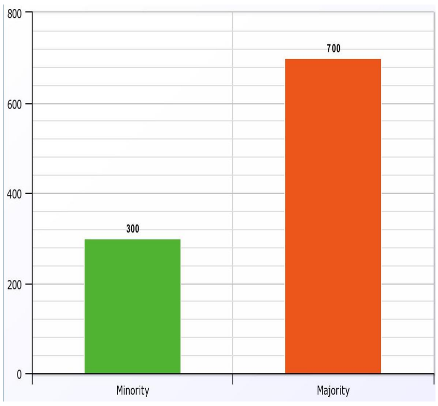
D2. Resampled



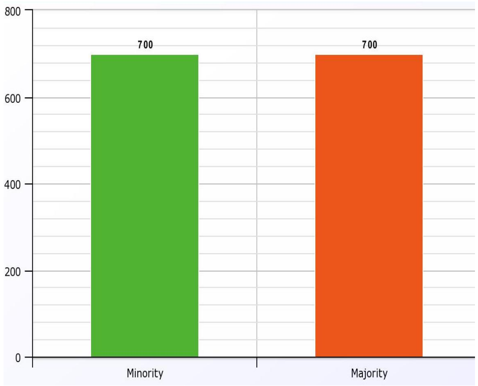
D3. Original



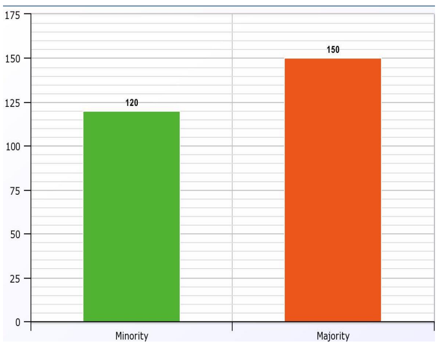
D3. Resampled



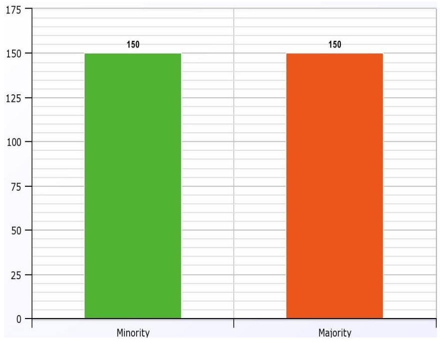
D4. Original



D4. Resampled



D5. Original



D5. Resampled

Figure 4. The before and after synthesis with SMOTE on the datasets is showed in the below table.

3. Classification Models

The single classifier and meta-learning ensemble classification methods are used in this study for testing. K-Nearest Neighbor (KNN), Support Vector Machine (SVM) and Nave Bayes are among the single classifiers employed (NB). The Stacking ensemble used is with (SVM) and (NB) as meta-learners.

- KNN - This is a classification technique that calculates the shortest distance from the query location on the training data to find the KNN.

The steps for implementing a KNN model are as follows:

1. Gather information.
2. Change k's value to zero.
3. Determine the thesised class by iterating from 1 to the entire amount of training data points.

a) Calculate the distance between each row of training data and the test data. Because it
is the most widely used method, we will utilise Euclidean distance as our distance metric.
b) Sort the estimated distances by distance values in ascending order.
c) Get the first k rows of a sorted array.
d) Get the most common kind of these rows.
e) Return the predicted class.

Equation for Euclidean Distance:

$$d\ (x_i,y_j) = \sqrt{\sum_{r=1}^{n}(ar(xi) - ar(xj))^2} \tag{1}$$

where,

$d\ (x_i,x_j)$: Is the Euclidean Distance
(x_i): xi is the data record
(x_j): xj is the data record
(a_r): for 'r' data record
i,j: and values for I and j are 1,2,3….n

We can simply determine the category or class of a dataset with the help of K-NN. Consider the diagram below:

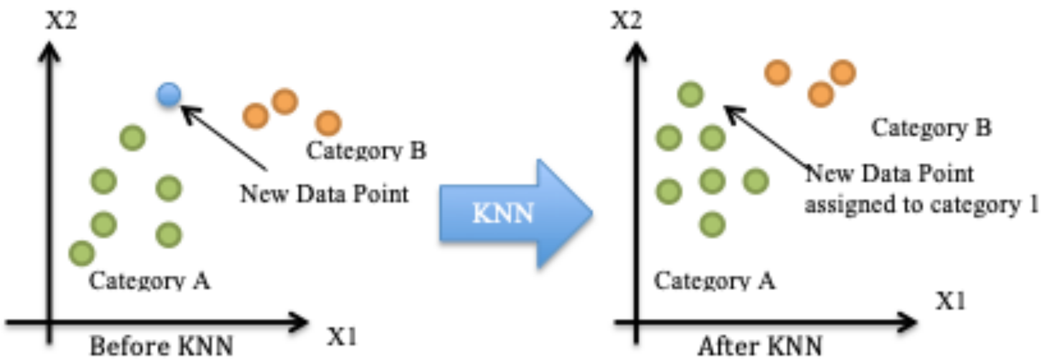


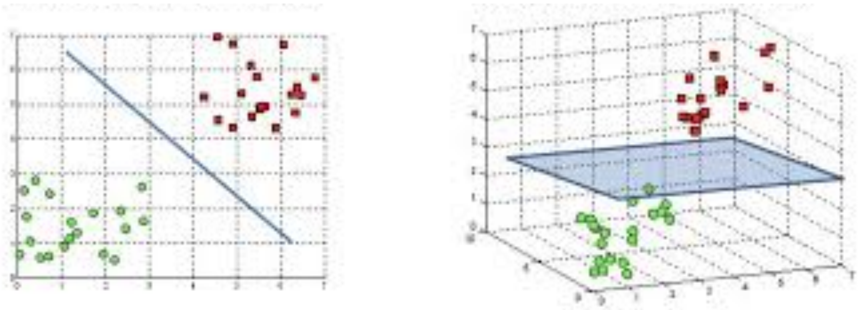
Figure 5. KNN working

• SVM - (Support Vector Machine) is a supervised classification method that alters data with the kernel before calculating the best border between likely outcomes. SVM stands for Support Vector Machine and is a supervised machine learning technique that can be used to handle classification and regression problems. However, it is mostly employed to solve classification issues.

In this technique, which depicts each data item as a point in n-dimensional space, the value of each feature is the value of a certain coordinate (where n is the number of characteristics you have). Then we locate the hyper-plane that best distinguishes the two classes to complete classification.

➢ Hyperplane and Support Vector:

Hyperplanes are decision boundaries that help categorise data. Data points on either side of the hyperplane can be allocated to different classifications.



A hyperplane in 2-D is a line   A hyperplane in 3-D is a plane

Figure 6. Hyperplane in SVM

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. We use these support vectors to increase the classifier's margin. If the support vectors are removed, the hyperplane's location will change. These are the considerations that will help us build our SVM.

Kernel SVM:

Kernel SVM takes a kernel function from the SVM algorithm and changes it into the appropriate format for mapping data to a higher, separable dimension.
Types of Kernel functions are:

$$K(X_i, X_j) = \left\{ \begin{array}{ll} X_i . X_j & \text{Linear} \\ (yX_i . X_j + C)^d & \text{Polynomial} \\ \exp(-y|X_i - X_j|^2) & \text{RBF} \\ \tanh(yX_i . X_j + C) & \text{Sigmoid} \end{array} \right\}$$

Types:

1. SVM (linear)
2. The degree of the polynomial should be given in Polynomial Kernel. Curved lines can be drawn in the input space.
3. It is utilised for non-linearly separable variables in the RBF Kernel. The metric squared Euclidean distance is used to calculate distance. Using a typical value for the parameter can cause our data to be overfit.
4. For this, a sigmoid kernel, which is similar to logistic regression, is used.

The kernel trick works by shifting data into a higher-dimensional feature space, allowing for linear classification separation.

- Nave Bayes - is a Bayes Theorem based probabilistic classification algorithm. It's a set of algorithms in which each pair of features is classified separately. The formula it employs is as follows:

$$\mathrm{P}(A/B) = \mathrm{P}(B/A)\,\frac{P(A)}{P(B)} \tag{2}$$

Where,

P(A) is the probability of hypothesis A being true. This is known as the prior probability.
P(B) is the probability of the evidence(regardless of the hypothesis).
P(B|A) is the probability of the evidence given that hypothesis is true.
P(A|B) is the probability of the hypothesis given that the evidence is there.

The Naive Bayes classifier makes the assumption that all of the features are unrelated. The presence or absence of one trait has no bearing on the presence or absence of another. We test a hypothesis given numerous pieces of evidence in real-world datasets (feature). As a result, computations become more difficult. To make things easier, the feature independence technique is utilised to ‘uncouple' various pieces of evidence and consider them as separate entities.

The assumptions that different naïve Bayes classifiers make about the distribution of P(A | B) are what distinguishes them.

One of these classifiers is: Gaussian Naive Bayes classifier

In Gaussian Naive Bayes, continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution. A Gaussian distribution is another name for a normal distribution. When plotted, it provides a bell-shaped curve that is symmetric about the mean of the feature values, as shown below:

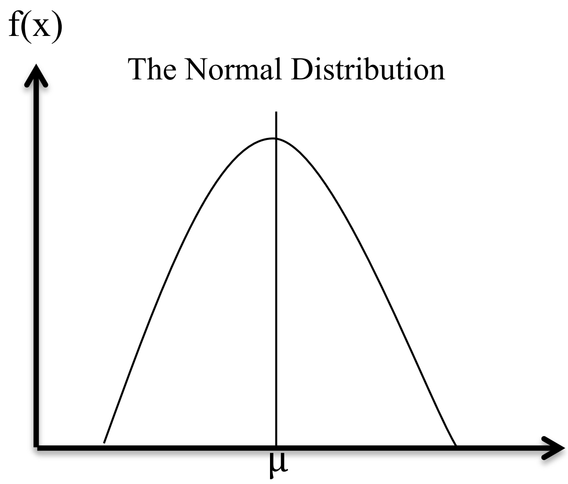


Figure 7. Gaussian function

4. Evaluation Matrices

These are the measures used to evaluate the performance of various classification methods when dealing with unbalanced datasets. Both assessing classification performance and directing classifier modelling, evaluation methods are critical. For these purposes, precision has traditionally been the most often used metric. When working with skewed datasets, it becomes clear that expected accuracy is biased toward the majority and is particularly sensitive to data distribution. Minority class misclassification has a far higher mistake rate than majority class misclassification, and minority class instances are less likely to be predicted than majority class instances.

- Confusion Matrix

After applying classification methods, the confusion matrix consists of rows that represent actual class and columns that represent the anticipated class. TN stands for true negative and refers to the number of negative events accurately categorised as negative. FP stands for false positive and the amount of negative events that were mistakenly labelled as positive, False negative is referred to as FN which is the number of events that are appropriately classified as positive. TP stands for true positive, and it relates to the number of positive situations that were mistakenly labelled as negative.

Table III. Confusion Matrix

	Predicted as Positive	Predicted as Negative
Actually Positive	True Positives (TP)	False Negatives (FN)
Actually Negative	False Positives (FP)	True Negatives (TN)

- Evaluation Measures

we use the following evaluation measures in our study - Accuracy, G-mean (geometric mean), F-measure and ROC (AUC).

a. Accuracy is the ratio of TP to TN i.e. the value of data points, which are correctly classified after classification. The term "accuracy" refers to how many data points are successfully anticipated. It is the most basic type of evaluation metric. The accuracy score is calculated as the number of correct points divided by the total number of points.

$$\text{Accuracy} = \frac{TP+TN}{(TP+TN+FP+FN)} \tag{3}$$

b. G-mean stands for geometric mean and is the performance measure that uses the product of specificity and sensitivity values. The geometric mean, or G-Mean, is a measure that combines sensitivity and specificity into a single value that balances both objectives.

$$\text{G-mean} = \sqrt{Specificity * \ Sensitivity} \tag{4}$$

c. F-measure is also known f-score and is mostly calculated for binary classification and uses the values of precision recall. The F-measures, on the other hand, do not account for true negatives. When there are more than two classes in a classification problem, the F-score is also used (Multiclass classification).

$$\text{F-measure} = \frac{TP}{TP+\frac{1}{2}(FP+FN)} \tag{5}$$

d. ROC is the Receiver Operating Characteristics, used to construct the curve from the FN and TP rates. The trade-off between the true positive rate and the false positive rate for a predictive model utilising different probability thresholds is summarised by ROC Curves. ROC AUC is the area under the ROC curve, which is calculated by taking average of TP and TN rate. The roc_auc_score() function can be used to compute the ROC's AUC.

$$\text{ROC AUC} = \frac{TPrate+TNrate}{2} \tag{6}$$

CHAPTER 4

EXPERIMENTAL SETUP

1. Dataset Analysis

The datasets used in this thesis are available in CSV format from the UCI machine repository.

UCI: The UCI Machine Learning Repository is a repository of databases, domain theories, and data generators used by the machine learning community to empirically evaluate machine-learning algorithms. Students, instructors, and researchers use it as a key source of machine learning data sets all across the world. The archive has been mentioned over 1000 times, indicating its importance. As a service to the machine learning community, it currently maintains 488 data sets.

Training Dataset: A training dataset is a set of data that we utilise to train our machine-learning model.

Testing Dataset: This is a dataset that we use to test the correctness of our model but not to train it. It could be referred to as the validation dataset. We may be required to gather instances in order to construct our datasets, or we may be provided a finite dataset that must be divided into sub-datasets.

Data Type: They can have a category or ordinal value, and they can be real or integer-valued. Strings, dates, timings, and other more complicated types can exist, but when using classic machine learning algorithms, they are often converted to real or categorical values.

Input Data: CSV File
CSV (Comma Separated Values) is a straightforward file format for storing tabular data in spreadsheets or databases. CSV files are plain text files that include tabular data (numbers and text). A data record is each line of the file. One or more fields, separated by commas, make up each record. The name for this file format comes from the fact that it uses a comma as a field separator.

In this study, we used five real-world datasets, which are listed in the table below by the total number of instances, the number of majority instances, the number of minority instances, the Imbalance ratio (IR), and the number of attributes used are showed below:

Pima Indian Diabetes Dataset - The class feature has these variables: '0' for negative diabetes cases and '1' for positive diabetes cases in the Pima Indian Diabetes Dataset. In the sample, there are 268 minority class positive diabetes cases and 500 majority class cases.

Wisconsin Breast Cancer Dataset - A benign (non-cancerous) sample is denoted by the class variable "2," while a malignant (cancerous) sample is denoted by the class variable "4". In the dataset, there are 241 minority class occurrences for benign class and 458 majority class instances for malignant class.

TABLE I. Datasets

Datasets	Description	Total Instances	Attributes	Minority Class	Majority Class	IR (Imbalance Ratio)
D1	Pima India Diabetes	768	9	268	500	0.53
D2	Breast Cancer Wisconsin	699	11	241	458	0.52
D3	Statlog Heart	270	14	120	150	0.80
D4	US Credit	1000	25	300	700	0.42
D5	Spam Base	4601	58	1813	2788	0.65

Statlog Heart Dataset - The class attributes are represented as '1' [heart disease absence] and '2' [heart disease presence] in the Statlog Heart Dataset. In the dataset, there are 120 minority class occurrences and 150 majority class occurrences.

US Credit Dataset - The class attribute is represented by two class variables, with '1' indicating a good class and '2' indicating a bad class in the US Credit Dataset. There are 700 instances of the majority class labelled Bad and 300 instances of the minority class labelled Good in the dataset.

Spam Base Dataset – Base Dataset for Spam – The class attributes are represented by two class variables: '1' for spam and '0' for solicited email. The collection contains 2788 majority cases of solicited email and 1813 minority occurrences of spam. majority cases of solicited email and 1813 minority cases of spam in the dataset.

2. Hardware and Software Requirements for Thesis:

Software Requirements:

Operating System: Windows 10, MacOs
Coding platform: Google Colaboratory, Kaggle Notebooks
Coding Language: Python

Google Colaboratory: Developed by Google, Colaboratory, or 'Colab' for short, allows you to write and execute Python in your browser, with no additional software required.

- Zero configuration required
- Free access to GPUs
- Easy sharing

Colab notebooks combine executable code and rich text, as well as graphics, HTML, LaTeX, and other elements, into a single document. When you create a Colab notebook, it is saved in your Google Drive account. You may easily share your Colab notebooks with coworkers or friends, allowing others to add comments or even edit them. Jupyter notebooks hosted by Colab are known as Colab notebooks. You may use Colab to analyse and visualize data using the full power of popular Python modules.

You may import data from your Google Drive account, including spreadsheets, as well as GitHub and a variety of other sources, into Colab notebooks. You can import an image dataset, train an image classifier on it, and test the model with Colab in just a few lines of code. Colab notebooks execute code on Google's cloud servers, so you can use Google hardware like GPUs and TPUs regardless of your machine's capabilities. To get started, all you need is a browser.

Kaggle Notebooks: These are the Jupyter notebooks developed by Kaggle, made up of a series of cells, each of which is formatted in either Markdown (for writing text) or your preferred programming language (for writing code). To begin a notebook, select "Notebook" from the "Create Notebook" menu. This will bring up the editing interface for Notebooks. R or Python can be used to create notebooks.

Python: Python is a programming language that is widely used because of its extensive capabilities, applicability, and ease of use. Python is the greatest programming language for machine learning because of its independent platform and popularity among programmers. Python is a high-level computer language for general-purpose programming that is interpreted. Python has a dynamic type system and memory management that is automated. It features a big and extensive standard library and supports several programming paradigms, including object-oriented, imperative, functional, and procedural.

Python has the following virtues:

a. Platform-independent
b. Simplicity and consistency
c. Variety of frameworks and libraries

Python libraries used in the thesis:

- Numpy - Numpy is a popular Python module that lets you process massive multi-dimensional arrays and matrices with a variety of high-level mathematical functions. It's useful for Machine Learning's basic scientific computations. It's great for linear algebra, the fourier transform, and generating random numbers.

- Pandas - Pandas is a well-known Python data analysis toolkit. The dataset must be prepared before training, as we all know. Pandas come in helpful in this scenario because it was designed expressly for data extraction and preparation. It provides high-level data structures as well as a comprehensive range of data analysis capabilities. It has a lot of built-in data grouping, combining, and filtering capabilities.

- Matplotlib - Matplotlib is a well-known Python data visualisation package. It's very useful when a coder needs to see how data patterns are represented. It's a 2D plotting library for making graphs and plots in 2D space. Python's pyplot package makes charting simple for programmers by allowing them to customize line styles, font characteristics, and axes formatting, among other things. It includes a variety of graphs and plots for data visualisation, including histograms, error charts, and bar charts, among others.

- Seaborn - Seaborn is a Python data visualisation Programme based on Matplotlib. It offers a high-level interface that allows you to create aesthetically beautiful and informative statistical graphics. Seaborn aids in data exploration and comprehension. Its graphing capabilities work with data frames and arrays containing full datasets, internally executing the necessary semantic mapping and statistical aggregation to produce effective graphs. Its declarative, dataset-oriented API lets you focus on the meaning of charts rather than the mechanics of generating them.

- Sklearn - Scikit-learn is undoubtedly Python's most helpful machine-learning library. Classification, regression, clustering, and dimensionality reduction are just a few of the useful capabilities in the sklearn toolkit for machine learning and statistical modelling.

<u>Hardware Requirements:</u>

Machine: Pentium 4
Hard disk: 10GB
RAM: 256MB

CHAPTER 5

RESULTS OBTAINED

Breast cancer data

```
data = pd.read_csv('../input/breastcancer/data.csv')

data.head()
```

[6]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mea
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30(
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.086
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.197
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.241
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.198

5 rows × 33 columns

Figure 8. Breast cancer data description

[23]:

```
feat_imp = pd.Series(mod.feature_importances_, index = selected_features)
feat_imp.nlargest(15).plot(kind = 'bar')
plt.title('Feature Importance')
plt.show()
```

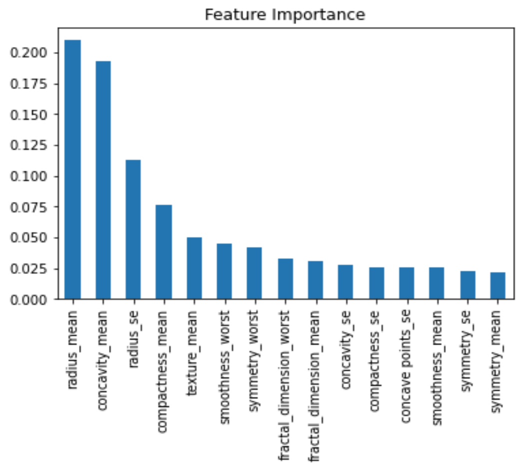


Figure 8.1

```
In [6]:  y.value_counts() # 0 = Benign   1 = Malignant
Out[6]:
0    357
1    212
Name: diagnosis, dtype: int64
```

Figure 8.2

```
[7]: <function matplotlib.pyplot.show(close=None, block=None)>
```

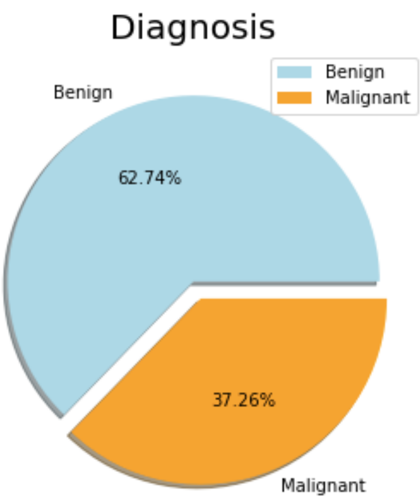


Figure 8.3

```
In [37]:
svm = SVC(random_state=0)
clfsvm = GridSearchCV(svm, param_grid = {'C': np.arange(0.1, 10, 0.2), 'kernel':
['rbf', 'linear', 'poly']}, cv = 5, scoring = 'recall')
clfsvm.fit(X_train, y_train)
print('''Recall score on training set is : {}
with parameters as: {}'''.format(clfsvm.best_score_, clfsvm.best_params_))
```

```
Recall score on training set is : 0.9264367816091955
with parameters as: {'C': 3.3000000000000007, 'kernel': 'linear'}
```

Figure 8.4

In [38]:
```
svmf = SVC(C = 6.7, kernel = 'linear', random_state=0)
svmf.fit(X_train, y_train)
y_predsvm = svmf.predict(X_test)
print('Recall score on test data for svm is: {}'.format(recall_score(y_test, y_predsvm)))
recall_svm = recall_score(y_test, y_predsvm)
```

```
Recall score on test data for svm is: 0.9365079365079365
```

Figure 8.5

In [12]:
```
import numpy as np
kernels = ['rbf', 'linear']
C = [1,10,20]
avg_scores = {}
for kval in kernels:
    for cval in C:
        cv_scores = cross_val_score(svm.SVC(kernel=kval,C=cval,gamma='auto'),x,y, cv=5)
        avg_scores[kval + '_' + str(cval)] = np.average(cv_scores)

print(avg_scores)
```

```
{'rbf_1': 0.6274181027790716, 'rbf_10': 0.6274181027790716, 'rbf_20': 0.6274181027790716, 'l
inear_1': 0.9455364073901569, 'linear_10': 0.9508150908244062, 'linear_20': 0.95081509082440
62}
```

Figure 8.6

```
scores = []

for model_name, mp in model_params.items():
    clf =  GridSearchCV(mp['model'], mp['params'], cv=5, return_train_score=False)
    clf.fit(x,y)
    scores.append({
        'model': model_name,
        'best_score': clf.best_score_
    })

df = pd.DataFrame(scores,columns=['model','best_score'])
df
```

25...

	model	best_score
0	SVM	0.950815
1	KNN	0.949045
2	GaussianNB	0.938519

Figure 8.7

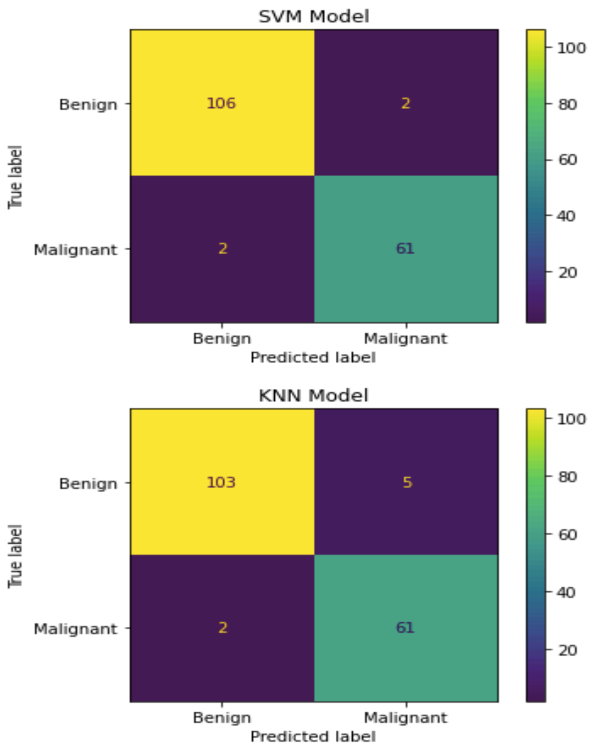


Figure 8.8

US Credit data

```
df = pd.read_csv('../input/creditcard.csv')
df.head()
```

[3]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V1
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	0.090794	-0.551600	-0.617801	-0.99139
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	-0.166974	1.612727	1.065235	0.48909
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	0.207643	0.624501	0.066084	0.71729
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	-0.054952	-0.226487	0.178228	0.50775
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	0.753074	-0.822843	0.538196	1.34585

Figure 9. US Credit data description

```
# we make a list ok k values for k-nearest- neighbours and through iterating over th
k_values = [1, 2, 3, 4, 5, 6, 7]
for k in k_values:
    over = SMOTE(sampling_strategy=0.1, k_neighbors=k)

    steps = [('over', over), ('under', under), ('model', model)]
    pipeline = Pipeline(steps=steps)
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
    scores = cross_val_score(pipeline, X, y, scoring='roc_auc', cv=cv, n_jobs=-1)
    score = mean(scores)
    print('> k=%d, Mean ROC AUC: %.3f' % (k, score))
```

```
> k=1, Mean ROC AUC: 0.909
> k=2, Mean ROC AUC: 0.906
> k=3, Mean ROC AUC: 0.909
> k=4, Mean ROC AUC: 0.911
> k=5, Mean ROC AUC: 0.910
> k=6, Mean ROC AUC: 0.911
> k=7, Mean ROC AUC: 0.914
```

Figure 9.1

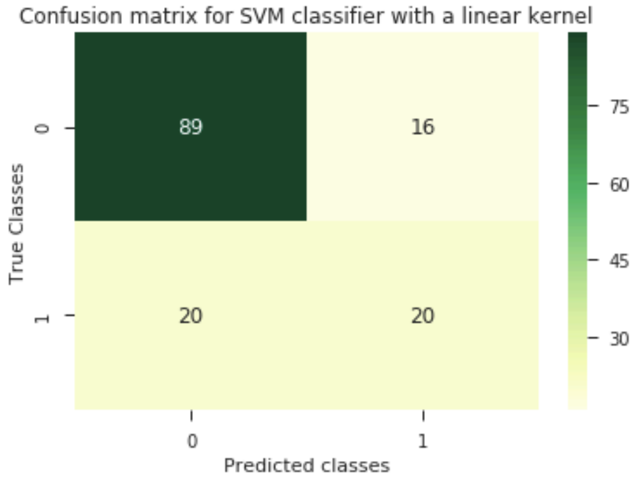
Pima India Diabetes data

```
data = pd.read_csv('../input/pimaindiansdiabetescsv/pima-indians-diabetes.csv')
data.head()
```

7]:

	6	148	72	35	0	33.6	0.627	50	1
0	1	85	66	29	0	26.6	0.351	31	0
1	8	183	64	0	0	23.3	0.672	32	1
2	1	89	66	23	94	28.1	0.167	21	0
3	0	137	40	35	168	43.1	2.288	33	1
4	5	116	74	0	0	25.6	0.201	30	0

Figure 10. Pima India Diabetes data description




```
test accuracy for Gaussian naive bayes classifier: 73.79 %
```

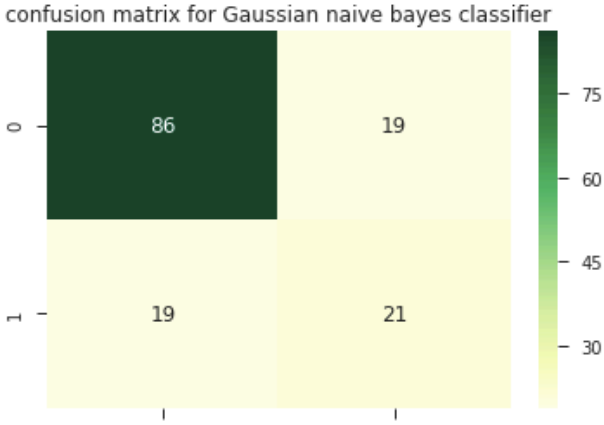


Figure 10.1

```
for clf, clf_name in zip(classifiers, classifier_names):
    cv_scores = cross_val_score(clf, train[features], train[target], cv=5)

    print(clf_name, ' mean accuracy: ', round(cv_scores.mean()*100, 3), '% std: ', round(cv_scores.var()*100, 3),
```

```
K nearest neighbors  mean accuracy:  71.153 % std:  0.07 %
SVM classifier with RBF kernel  mean accuracy:  63.904 % std:  0.0 %
SVM classifier with linear kernel  mean accuracy:  77.373 % std:  0.058 %
Gaussian Naive Bayes  mean accuracy:  75.99 % std:  0.143 %
```

Figure 10.2

Statlog Heart data

```
data = pd.read_csv('../input/heart-data/data.csv')
data.head
```

```
<bound method NDFrame.head of      age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
0     70    1   4       130   322    0        2      109      0      2.4
1     67    0   3       115   564    0        2      160      0      1.6
2     57    1   2       124   261    0        0      141      0      0.3
3     64    1   4       128   263    0        0      105      1      0.2
4     74    0   2       120   269    0        2      121      1      0.2
..   ...  ...  ..       ...   ...  ...      ...      ...    ...      ...
265   52    1   3       172   199    1        0      162      0      0.5
266   44    1   2       120   263    0        0      173      0      0.0
267   56    0   2       140   294    0        2      153      0      1.3
268   57    1   4       140   192    0        0      148      0      0.4
269   67    1   4       160   286    0        2      108      1      1.5

     slope  ca  thal  target
0        2   3     3       2
1        2   0     7       1
2        1   0     7       2
3        2   1     7       1
4        1   1     3       1
..     ...  ..   ...     ...
265      1   0     7       1
266      1   0     7       1
267      2   0     3       1
268      2   0     6       1
269      2   3     3       2
```

Figure 11. Statlog Heart data description

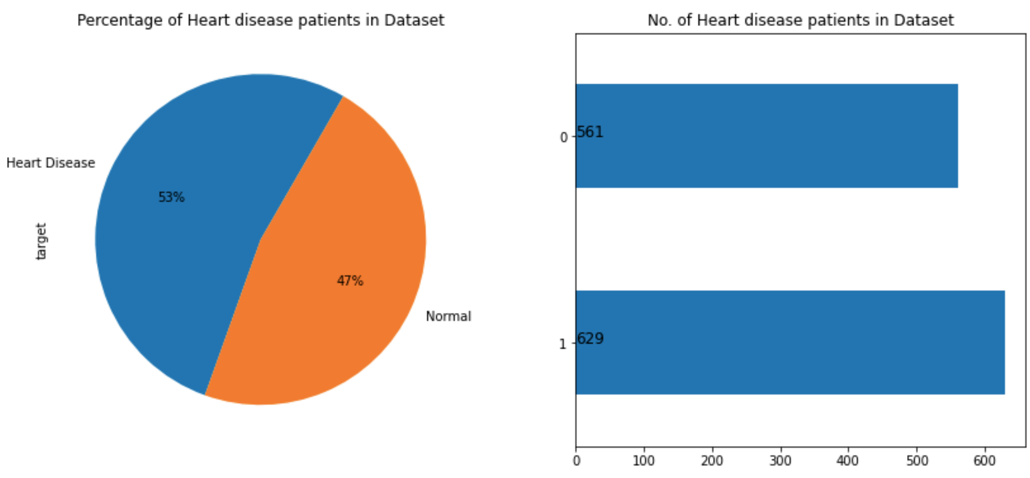


Figure 11.1

[51…

	Model	Accuracy	Precision	Sensitivity	Specificity	F1 Score	ROC	Log_Loss	mathew_corrcoef
0	STacked Classifier	0.885106	0.869231	0.918699	0.848214	0.893281	0.883457	3.968343	0.770445

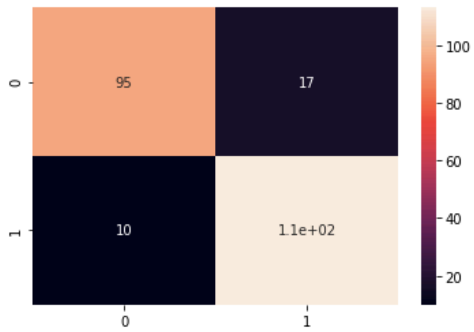


Figure 11.2

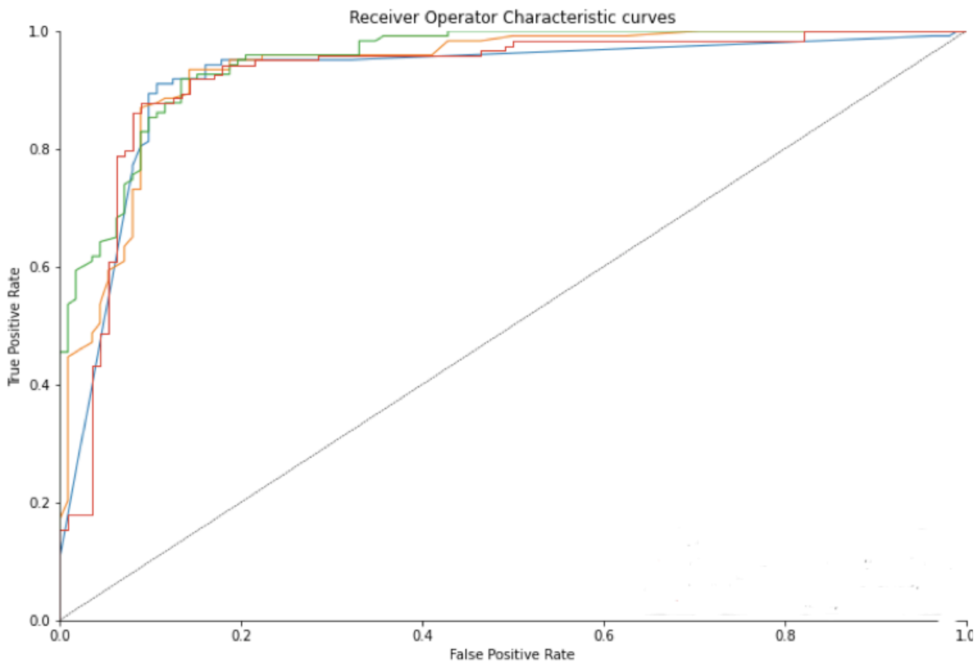


Figure 11.3

[21]:

```
sns.scatterplot(x = 'resting_blood_pressure', y = 'cholesterol', hue = 'target', data = dt)
```

[21… <matplotlib.axes._subplots.AxesSubplot at 0x7f0c281b77b8>

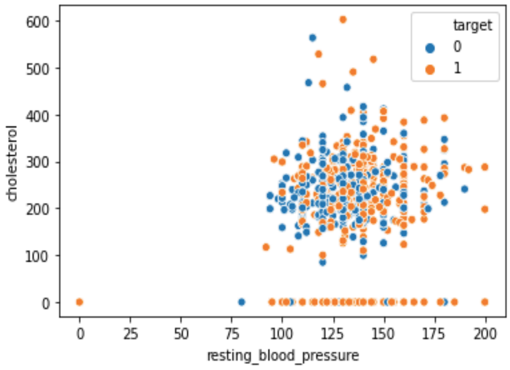


Figure 11.4

Spam Base data

```
data = pd.read_csv('../input/spam-mails-dataset/spam_ham_dataset.csv'
data.head
```

```
<bound method NDFrame.head of       Unnamed: 0 label
0            605   ham  Subject: enron methanol ; meter # : 988291\r\n...
1           2349   ham  Subject: hpl nom for january 9 , 2001\r\n( see...
2           3624   ham  Subject: neon retreat\r\nho ho ho , we ' re ar...
3           4685  spam  Subject: photoshop , windows , office . cheap ...
4           2030   ham  Subject: re : indian springs\r\nthis deal is t...
...          ...   ...                                                ...
5166        1518   ham  Subject: put the 10 on the ft\r\nthe transport...
5167         404   ham  Subject: 3 / 4 / 2000 and following noms\r\nhp...
5168        2933   ham  Subject: calpine daily gas nomination\r\n>\r\n...
5169        1409   ham  Subject: industrial worksheets for august 2000...
5170        4807  spam  Subject: important online banking alert\r\ndea...

      label_num
0             0
1             0
2             0
3             1
4             0
...         ...
5166          0
5167          0
5168          0
5169          0
5170          1

[5171 rows x 4 columns]>
```

Figure 12. Spam Base data description

```
### Final Testing with test data
fin_clf = grid_search.best_estimator_
fin_clf.fit(X_train_dtm, y_train)
print(f"Test Scores: {fin_clf.score(X_test_dtm, y_test)}")

y_pred = fin_clf.predict(X_test_dtm)
print(f"Accuracy: {accuracy_score(y_pred, y_test)}")
```

```
Test Scores: 0.9802513464991023
Accuracy: 0.9802513464991023
```

Figure 12.1

```
svc = SVC(C=1.0,kernel='linear',gamma='auto')
# C here is the regularization parameter. Here, L2 penalty is used(default). It is the inverse of t
# As C increases, model overfits.
# Kernel here is the radial basis function kernel.
# gamma (only used for rbf kernel) : As gamma increases, model overfits.
svc.fit(train_x,train_y)
y_pred2 = svc.predict(test_x)
print("Accuracy Score for SVC : ", accuracy_score(y_pred2,test_y))
```

```
Accuracy Score for SVC :  0.9164733178654292
```

Figure 12.2

```
[9]:
mnb = MultinomialNB(alpha=1.9)         # alpha by default is 1. alpha must always be > 0.
# alpha is the '1' in the formula for Laplace Smoothing (P(words))
mnb.fit(train_x,train_y)
y_pred1 = mnb.predict(test_x)
print("Accuracy Score for Naive Bayes : ", accuracy_score(y_pred1,test_y))
```

```
Accuracy Score for Naive Bayes :  0.9296210363495746
```

Figure 12.3

Table IV. Accuracy Score

Accuracy						
Algorithms	**Cases**	**Datasets**				
		D1	**D2**	**D3**	**D4**	**D5**
SVM	None	0.775	0.970	0.852	0.788	0.908
	Smote	0.692	0.975	0.846	0.718	0.896
Naïve bayes (NB)	None	0.763	0.960	0.852	0.766	0.795
	Smote	0.723	0.969	0.831	0.714	0.844
KNN	None	0.823	0.978	0.833	0.807	0.931
	Smote	0.796	0.982	0.815	0.762	0.923
Stacking (SVM – NB)	None	0.821	0.971	0.851	0.788	0.931
	Smote	**0.856**	**0.969**	**0.880**	**0.810**	**0.941**

Table V. F-Measure

F-Measure						
Algorithms	**Cases**	**Datasets**				
		D1	**D2**	**D3**	**D4**	**D5**
SVM	None	0.623	0.957	0.831	0.592	0.877
	Smote	0.664	0.976	0.856	0.620	0.903
Naïve bayes (NB)	None	0.645	0.944	0.829	0.578	0.786
	Smote	0.696	0.970	0.853	0.634	0.875
KNN	None	0.728	0.969	0.812	0.642	0.911
	Smote	0.780	0.982	0.841	0.698	0.930
Stacking (SVM – NB)	None	0.731	0.969	0.856	0.655	0.921
	Smote	**0.800**	**0.975**	**0.890**	**0.741**	**0.951**

Table VI. G-Mean

G-Mean						
Algorithms	**Cases**	**Datasets**				
		D1	**D2**	**D3**	**D4**	**D5**
SVM	None	0.695	0.968	0.848	0.682	0.894
	Smote	0.699	0.975	0.841	0.672	0.902
Naïve bayes (NB)	None	0.720	0.962	0.847	0.679	0.813
	Smote	0.718	0.968	0.840	0.682	0.815
KNN	None	0.782	0.978	0.830	0.723	0.925
	Smote	0.794	0.982	0.822	0.734	0.926
Stacking (SVM – NB)	None	0.801	0.981	0.850	0.721	0.931
	Smote	**0.850**	**0.990**	**0.866**	**0.788**	**0.961**

Table VII. ROC AUC

ROC Area						
Algorithms	**Cases**	**Datasets**				
		D1	**D2**	**D3**	**D4**	**D5**
SVM	None	0.719	0.968	0.848	0.710	0.896
	Smote	0.723	0.975	0.842	0.702	0.903
Naïve bayes (NB)	None	0.825	0.987	0.916	0.801	0.941
	Smote	0.823	0.988	0.908	0.748	0.945
KNN	None	0.896	0.998	0.931	0.874	0.984
	Smote	0.902	0.997	0.929	0.877	0.983
Stacking (SVM – NB)	None	0.900	0.968	0.931	0.871	0.984
	Smote	**0.951**	**0.998**	**0.945**	**0.901**	**0.988**

The results obtained are recorded as follows: highest values are highlighted in bold.

CHAPTER 8

CONCLUSION & FUTURE WORK

Due to the presence of CIP in the data, the classifier models perform less than optimal. The presence of majority instances mislead the accuracy and other performance measures of the classifier model. So, the focus of this thesis is to address the problem of class imbalance.

We use two Hybrid method in this thesis: an oversampling technique – SMOTE (for the dataset), and an ensemble method called Stacking for the classifier level. We balanced the five datasets - Pima India Diabetes (PID), Breast Cancer Wisconsin, Statlog Heart, US Credit, and Spam Base - using the SMOTE method, and then used Stacking on the base classifiers SVM (Support Vector Machine), KNN (K-Nearest Neighbor) and Nave Bayes to get the performance measures - Accuracy, F-measure, G-mean, and Roc Auc. Based on the results, we discovered that stacking combined with Smote outperforms the other base classifiers.

Based on the research, we can say that the problem of class imbalance in the data is an important measure for classifying the data. The result outcomes of the thesis shows that Stacking algorithm along with SMOTE technique gives better performance than the other base models with or without smote, as not only the Accuracy score is improved but also F-Measure, G-mean and ROC AUC is also showing better results. So, we conclude that Class imbalance problem can be resolved by this proposed dual approach and can solve the imbalance distribution in the real world data.

The dataset used in this thesis is of binary classification, which is however a limitation, so in the further study we can test for solution for the imbalance class problem in multiclass datasets.

REFERENCES

[1]. K. Yang *et al.*, “Hybrid Classifier Ensemble for Imbalanced Data,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. PP, pp. 1–14, 2019.

[2]. Vaishali, G., “An Overview Of Classification Algorithms For Imbalanced Datasets”, Int Journal of Emerging technology and Advanced Engineering, 2(4), 2012.

[3]. G. Hu, T. Xi, F. Mohammed, and H. Miao, “Classification of wine quality with imbalanced data,” *Proc. IEEE Int. Conf. Ind. Technol.*, pp. 1712–1717, 2016.

[4]. S. Ahmed, A. Mahbub, F. Rayhan, R. Jani, S. Shatabda, and D. M. Farid, “Hybrid Methods for Class Imbalance Learning Employing Bagging with Sampling Techniques,” *2nd Int. Conf. Comput. Syst. Inf. Technol. Sustain. Solut. CSITSS 2017*, pp. 1–5, 2018.

[5]. Y. Yang, P. Xiao, Y. Cheng, W. Liu, and Z. Huang, “Ensemble Strategy for Hard Classifying Samples in Class-Imbalanced Data Set,” *Proc. - 2018 IEEE Int. Conf. Big Data Smart Comput.BigComp 2018*, pp. 170–175, 2018.

[6]. K. UlagaPriya and S. Pushpa, "A Comprehensive Study on Ensemble-Based Imbalanced Data Classification Methods for Bankruptcy Data," 2021 6th International Conference on Inventive Computation Technologies (ICICT), 2021, pp. 800-804, doi: 10.1109/ICICT50816.2021.9358744.

[7]. C. Jia, M. Zhang, C. Fan, F. Li and J. Song, "Formator: predicting lysine formylation sites based on the most distant undersampling and safe-level synthetic minority oversampling," in IEEE/ACM Transactions on Computational Biology and Bioinformatics, doi: 10.1109/TCBB.2019.2957758.

[8]. P.J.Huang, “Classification of Imbalanced Data Using Synthetic Over – Sampling Techniques,”University of California,2015.

[9]. N. V Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE : Synthetic Minority Over-sampling Technique,” J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.

[10] H. He, Y. Bai, E. A. Garcia, and S. Li, “ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning,” in Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, 2008, no. 3, pp. 1322–1328.

[11]. H.Han, W.Wang, and B.Mao,“Borderline - SMOTE A New Over Sampling Method in,” in International Conference on Intelligent Computing, ICIC 2005, Hefei, China, August 23-26, 2005, Proceedings, Part I, Springer Berlin Heidelberg, 2005, pp. 878–887.

[12]. T. Fahrudin, J. L. Buliali, and C. Fatichah, “RANDSHUFF: An algorithm to handle imbalance class for qualitative data,” Int. Rev. Comput. Softw., vol. 11, no. 12, pp. 1093 1104, 2016.

[13]. N. Chanamarn, K. Tamee, and P. Sittidech, “Stacking technique for academic achievement prediction,” *Int. Work. Smart Info-Media Syst. Asia (SISA 2016)*, no. Sisa 2016, pp. 14–17, 2016.

[14]. C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, “Safe-Level- SMOTE: Safe Level-Synthetic MI Over-Sampling Technique for handling the Class Imbalance Problem”, PADD2009, Springer LNAI, Vol. 5476, pp. 475-482, 2009.

[15]. Y. Pristyanto, A. F. Nugraha, I. Pratama, A. Dahlan and L. A. Wirasakti, "Dual Approach to Handling Imbalanced Class in Datasets Using Oversampling and Ensemble Learning Techniques," 2021 15th International Conference on Ubiquitous Information

Management and Communication (IMCOM),2021, pp. 1-7, doi10.1109/IMCOM51814.2021.9377420.
[16]. K. Yang et al., "Progressive Hybrid Classifier Ensemble for Imbalanced Data," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, doi: 10.1109/TSMC.2021.3051138.
[17]. A. Gosain and S. Sardana, "Handling class imbalance problem using oversampling techniques: A review," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017, pp. 79-85, doi: 10.1109/ICACCI.2017.8125820.
[18]. Maheshwari, S., Jitendra Agrawal and S. Sharma. “A New approach for Classification of Highly Imbalanced Datasets using Evolutionary Algorithms.” (2011).
[19]. N. V. Chawla et al., “SMOTE:Synthetic Minority Over Sampling Technique”, Journal of Artificial Intelligence Research, Vol. 16, pp 321- 357, 2002.
[20]. N. V. Chawla, ‘‘Data mining for imbalanced datasets: An overview,’’ Data Mining and Knowledge Discovery Handbook. Springer, pp. 853–867, 2005.
[21]. T. Jo and N. Japkowicz, ‘‘Class imbalances versus small disjuncts’’, ACM SIGKDD Explorations Newslett., vol. 6, no. 1, pp. 40–49, 2004.
[22]. J. A. Saez et al., “Managing Borderline and Noisy examples in Imbalanced Classification by combining SMOTE with Ensemble Filtering”, IDEAL2014, Springer LNCS, Vol. 8669, pp. 61-68, 2014.B.X. Wang and N. Japkowicz, “Imbalanced Data Set Learning with Synthetic Samples”, Proc. IRIS Machine Learning Workshop, 2004.
[23]. H. He, Y. Bai, E. A. Garcia, and S. Li, ‘‘ADASYN: Adaptive synthetic sampling approach for imbalanced learning,’’ in Proc. IEEE Int. Joint Conf. Neural Netw., IEEE World Congr. Comput. Intell., pp. 1322– 1328, Jun. 2008.
[24]. H. Han, W. Wang, and B. Mao, “Borderline-SMOTE: A new oversampling method in Imbalanced Data-sets Learning”, In. ICIC 2005.LNCS, Springer, Heidelberg, Vol. 3644, pp. 878-887, 2005.
[25]. C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, “Safe-Level SMOTE: Safe Level- Synthetic MI Over-Sampling Technique for handling the Class Imbalance Problem”, PADD2009, Springer LNAI, Vol. 5476, pp. 475-482, 2009.
[26] Guo, X., et al. “On the class imbalance problem”, Natural Computation, ICNC'08. IEEE, Fourth International Conference, 2008.
[27] Vaishali, G., “An Overview Of Classification Algorithms For Imbalanced Datasets”, Int Journal of Emerging technology and Advanced Engineering, 2(4), 2012.
[28] K. P. N. V. Satyashree, and J. V. R. Murthy, “An Exhaustive Literature Review on Class Imbalance Problem”, Int. Journal of Emerging Trends and Technology in Computer Science Vol. 2, No.3, pp 109-118, 2013.
[29] G. E. A. P. A Batista et al., “A study of the behaviour of several methods for balancing machine learning training data”, SIGKDD Expl. Newl., Vol 6, No. 1, pp 20-29, 2004.
[30] R. Malhotra and S. Kamal, "Tool to handle imbalancing problem in software defect prediction using oversampling methods," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017, pp. 906-912, doi: 10.1109/ICACCI.2017.8125957.
[31] P. Kaur and A. Gosain, "Density-oriented approach to identify outliers and get noiseless clusters in Fuzzy C — Means," International Conference on Fuzzy Systems, 2010, pp. 1-8, doi: 10.1109/FUZZY.2010.5584592.
[32] G. Zhou and F. Guo, "Research on Sampling Diversity Method in Ensemble Learning Base on Margin," 2019 International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), 2019, pp. 316-319, doi:

10.1109/MLBDBI48998.2019.00071.
[33] W. -Y. Lin and I. Dai, "An AdaBoost-Inspired Ensemble Method for ADR Signal Detection," 2020 IEEE International Conference on Healthcare Informatics (ICHI), 2020, pp. 1-2, doi: 10.1109/ICHI48887.2020.9374305.
[34] P. Kumkar, I. Madan, A. Kale, O. Khanvilkar and A. Khan, "Comparison of Ensemble Methods for Real Estate Appraisal," 2018 3rd International Conference on Inventive Computation Technologies (ICICT), 2018, pp. 297-300, doi: 10.1109/ICICT43934.2018.9034449.
[35] C. Perales-González, F. Fernández-Navarro, M. Carbonero-Ruz and J. Pérez-Rodríguez, "Global Negative Correlation Learning: A Unified Framework for Global Optimization of Ensemble Models," in IEEE Transactions on Neural Networks and Learning Systems, doi: 10.1109/TNNLS.2021.3055734.
[36] Y. Tian and X. Wang, "SVM ensemble method based on improved iteration process of Adaboost algorithm," 2017 29th Chinese Control And Decision Conference (CCDC), 2017, pp. 4026-4032, doi: 10.1109/CCDC.2017.7979205.
[37] Y. ZHANG, R. LU, J. HUANG and D. GAO, "Evolutionary-Based Ensemble Under-Sampling for Imbalanced Data," 2019 16th International Computer Conference on Wavelet Active Media Technology and Information Processing, 2019, pp. 212-216, doi: 10.1109/ICCWAMTIP47768.2019.9067647.