Prediction of Diabetes Induced Complications Using Different Machine Learning Algorithms

Supervised By:

Dr. Md. Ashraful Alam

Assistant Professor

Department of Computer Science and Engineering

BRAC University

Submitted By:

Tahsinur Rahman	15101128
Sheikh Mastura Farzana	15101077
Aniqa Zaida Khanom	15101106

Department of Computer Science and Engineering

BRAC University

Submitted on: August 02, 2018

DECLARATION

We, hereby, declare that this Thesis paper is based on the results conducted by ourselves. All the experiments, their respective results, including graphs, have been implemented over the past few months after proper research. Also, all the Reference materials have been correctly cited throughout the paper. This Thesis, neither in whole nor in part, has been previously submitted for any degree.

_____

Dr. Md. Ashraful Alam
Assistant Professor,
Department of Computer Science and Engineering,
BRAC University

_____

Tahsinur Rahman

_____

Sheikh Mastura Farzana

_____

Aniqa Zaida Khanom

TABLE OF CONTENTS

Page No.

LIST OF FIGURES

Page No.

LIST OF TABLES

Page No.

LIST OF EQUATIONS

Page No.

ABSTRACT

Machine Learning is an ever expanding field of Artificial Intelligence which uses huge amount of data to develop algorithms that can detect patterns and systems. One such application of Machine Learning is developing predictive models for disease prediction. On the other hand, in spite of huge advancements in Medical Science and discovery of complex diseases making everyone more health conscious, there is no way in Medical Science to predict prevalence of diseases. However, upon having relevant data Machine Learning methods can predict onset of many diseases. This paper presents the comparative analysis of different Machine Learning algorithms and their results in predicting the health complications related to Diabetes Mellitus. Diabetes Mellitus is a medical condition of the Pancreas in which the body's ability to produce or respond to the hormone, Insulin, diminishes. As a result, over time it damages other organs in the body- primarily Kidney, Liver, Eyes, Heart and Brain. Since in most cases the threats posed by Diabetes are not known before it is too late, hence it requires a great amount of consciousness in order to prevent onset of other related diseases. To this day, there is no prevention of Diabetes, since it is largely dependent on the genetics of a person. However, if a person is monitored closely it is possible to indicate Diabetes related complications. This proposed model uses time series data of a year that contains 164 features including results of different pathological tests. Methods such as Logistic Regression, SVM, Naïve Bayes, Decision Tree and Random Forest have been used in a supervised environment to predict the probability of Diabetes induced Nephropathy and Cardiovascular disease. PCA was applied beforehand to reduce the dimensionality of the dataset. Decision Tree without PCA produced the best results for Nephropathy with an AUC score of 0.87. While Naïve Bayes without PCA produced the best results for Cardiovascular disease, with an AUC score of 0.74. In summary, the model proposed in this paper predicts the risk of Nephropathy better than the risk of Cardiovascular disease.

Keywords: Diabetes Complications, PCA, missForest, SVM, Logistic Regression, Naïve Bayes, Decision Tree with AdaBoost.

1 INTRODUCTION

Machine Learning is the most advanced technique used today for pattern and decision rule extraction from a particular dataset. Despite being a branch of Artificial Intelligence, at its core, Machine Learning depends on statistical techniques. This has opened a new horizon for data modeling, data representation, data reasoning and data learning for contemporary computational science. Even though Artificial Intelligence could not quite stand up to all the promises it made when it first surfaced, Machine Learning is progressing forward at an astounding speed towards the previously promised direction. Scientists have already been able to form directly utilizable algorithms such as Neural Networks, Deep Learning, Classification and Association rules, Support Vector Machines (SVM) and Text Mining Pipelines using Machine Learning. Algorithms such as Decision Trees (DT), Naïve Bayes (NB), Logistic Regression (LR) and Random Forest (RF) have been derived using statistical theories and probability rules with Machine Learning. These algorithms are often used in embedded pipelines for extracting knowledge from data to derive decision rules and patterns. Machine Learning algorithms are being used in various prediction models such as weather prediction, sports result prediction, stock market prediction and to some extent medical condition prediction. However, use of Machine Learning for passive health condition prediction is still rare. Diabetes Mellitus is one such example of health condition. Diabetes is a genetics-dependent disease, which is divided into two broad categories based on occurrence of the disease; Type-1 patients are those who have had Diabetes from birth and it is likely that their pancreatic functions never developed properly in the first place while Type-2 patients are those who develop Diabetes over time as their Pancreas stops working properly owing to old age, excessive intake of glucose, etc.

1.1 MOTIVATION

According to IDF Atlas published in 2017, there are around 424.9 million Diabetes patients around the world aged from 20-79 years, of whom 95% suffer from Type 2 Diabetes Mellitus (T2DM). It is predicted that the number will increase to 628.6 million by 2045. Several Machine Learning based models exist that deal with Diabetes Mellitus. However, most of these systems only predict the probability of a person having Diabetes in the near future. Diabetes Mellitus can induce other complications like Nephropathy, Cardiovascular disease, Retinopathy and Diabetic

Foot disease. In 2017 alone, 4 million people died all around the world due to diabetic related complications, mostly because they were not monitored closely and warned beforehand. There is a scope to introduce a complete system that can correctly predict onset of complications caused by T2DM using Machine Learning techniques, which can save thousands, if not millions, of lives around the world. This influenced the research work done in this paper.

1.2 OBJECTIVES

Every year millions of people are dying not only because of Diabetes but also due to the Diabetes induced diseases and complications like Nephropathy, Neuropathy, Retinopathy, Cardiovascular disease and Diabetic Foot disease caused by it. This paper describes a predictive model built using Machine Learning algorithms. It can successfully predict probability of Nephropathy and Cardiovascular disease onset due to T2DM using an embedded pipeline comprising various Machine Learning algorithms. This is different to the conventional Diabetes predicting systems since it emphasizes on predicting Diabetes related complications. The pipeline has been applied upon a dataset containing pathological test results of 779 patients. This type of pipelines are often described as data mining model. The algorithms being used to implement the pipeline are Linear Regression (LR), Support Vector Machines (SVM), Naïve Bayes (NB), Random Forest (RF) Classifier, Decision Tree and Decision Tree with AdaBoost.

1.3 FUNDAMENTALS OF MACHINE LEARNING

Machine Learning is a branch of Artificial Intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. In other words, Machine Learning is teaching machines how to learn using Statistics and Probability.

Based on the different ways of teaching a machine to learn, Machine Learning divides the types of learning to a few major categories.

1. Supervised Learning uses a dataset as training examples with each example consisting of certain label/s which identify it. Using learning algorithms, the machine is taught to correctly identify new data given to it based on the previous dataset.
2. Unsupervised Learning does not give a definite output like supervised learning. Rather, it aims to find structures, patterns and trends in the given data.

3. Reinforcement Learning is a learning method that interacts with its environment by producing actions and discovers errors or rewards based on the action. This method allows the machine to determine the ideal behavior in a specific context by itself.

Through Machine Learning, it is possible to process and analyze massive amounts of data with proper training of the machine with required resources. This field of AI has presented a new wave of opportunities to develop systems for the betterment, including self-driving cars, practical speech recognition, effective web search, etc. in just the past decade. While it has brought about immense advancement in Genetic Engineering, in the field of Medical Science, Machine Learning has started to make progress very recently. Hence, there are great prospects in improving the infrastructures of Medical Science by means of using Machine Learning in disease prediction and identification, personalized treatment, drug discovery and manufacturing, etc.

1.4 THESIS ORIENTATION

The next section of the paper discusses about the similar works done before in the same field by different researches. Afterwards, the proposed model is described followed by a detailed description of the dataset and its features used in this project. Section five explains the system implementation including the process of missing value imputation, categorical variable conversion, feature scaling, Principal Component Analysis (PCA), class imbalance and Oversampling, train-test split and the description of the algorithms used for model implementation. The following section six includes the experimental results along with their performance metrics and discusses the results of the experiments through comparison between different models. Subsequently, the paper ends with the details of the challenges faced in this project and few concluding remarks.

2 LITERATURE REVIEW

Medical Science has advanced many folds in recent decades. Contribution of science, hard work from researchers and advanced technologies have made it possible to detect various diseases in their initial stages. Courtesy of Genetic Engineering, scientists can actually predict if a child is going to have any genetic diseases even before birth. Incredible breakthroughs have been made in the fields of medicine which has made it possible to cure many terminal diseases. Doctors

have found various new aspects in the field of Diabetes in past years [1]. Many research papers have been published discussing the factors which assist Diabetes development [2, 3]. Furthermore, numerous researches have been conducted to find the impact of Diabetes on health and the type of complications it may cause [4, 5]. Doctors have conducted tests on fixed set of people and have found out that Diabetes can induce Nephropathy, Neuropathy, Retinopathy and Cardiovascular diseases in the long run. However, most medical researches require high-end lab facilities, funding and volunteers to conduct experiments, and above all, time. On the other hand, Artificial Intelligence is becoming more accurate in pattern recognition and predictions. Recent development in Neural Network systems has made it even easier to teach a system how to solve a problem; there are many AI bots that can beat humans in games like chess, go, checkers etc. In July 2018 an AI system beat 15 Chinese doctors in a tumor detection competition [6] and this bot is not the first one to do such a thing. Machine Learning is the branch of AI that has successfully been used to generate predictive models for stock markets, weather condition, traffic condition, suitable habitat detection and many more. Nowadays, embedded pipeline of Machine Learning based algorithms is being used to predict onset of various diseases. In several papers, it is described how in the case of diseases like Cancer, Mental health and Cardiovascular conditions, scientists are applying predictive algorithms with satisfactory accuracy [7, 8, 9]. Abundance of Machine Learning models exist that can diagnose if a person has Diabetes or is prone to develop Diabetes [10]. However, models that can predict onset of Diabetes induced health complications are very rare. One such model used a data mining pipeline to predict T2DM related complications using electronic health record data [11]. Author Dagliati (2018) predicted complications such as Neuropathy, Nephropathy and Retinopathy with an accuracy of 0.83 [11]. The researchers had a complete dataset and used very few features to conduct the research which reduced complexity of the model. Another notable work done by Cho in 2008 explains a model which used visualization and feature selection to predict diabetic Nephropathy [12]. Apart from the above mentioned work, there is almost no other significant research that has directly addressed how Machine Learning can help predict T2DM induced health complications. Nonetheless, it is possible to develop a model that can accurately predict onset of Diabetes related complications. Machine Learning is an asset that has the ability to help doctors overcome their limitations by improving prediction and diagnosis of diseases.

3 PROPOSED MODEL

In this paper, several methods were used on a dataset consisting of 779 T2DM patients to determine the risk of Nephropathy and the risk of Cardiovascular disease. Figure 1 below represents the experimental work flow of the model in this project.

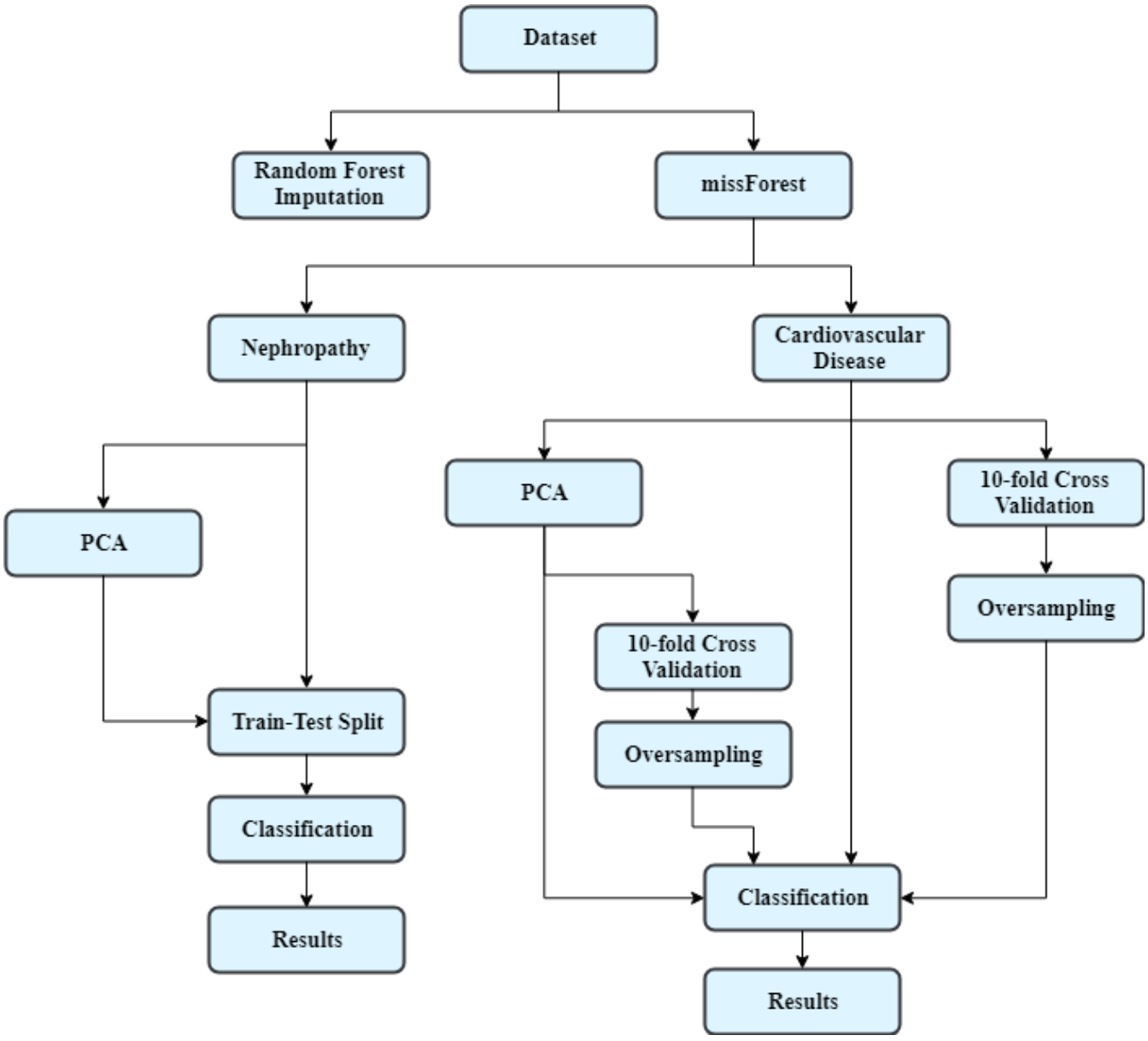


Fig. 1. Experimental Work Flow of the System.

At first, the data was cleaned by dropping instances with significant amount of missing values. Afterwards, mean/mode imputation was used to fill the missing values. However, the whole process was done manually and the results of mean/mode imputation were too poor, hence they were discarded. Instead, Random Forest and missForest algorithms were used to impute the missing values. These algorithms produced better results when it came to imputation of missing values. The categorical variables were then converted to numerical ones by introducing dummy variables. Since all the variables have different units and ranges, they were all brought to the same scale by apply a feature scaling technique called Standardization. After all the variables were taken to the same scale, a dimensionality reduction technique called Principal Component Analysis (PCA) was applied. This was done to convert the correlated variables to a smaller number of uncorrelated principal components. The problem at hand can be defined as a multi-label classification problem since a patient can have either of the complications or both. Hence, it was divided into two separate binary classification where risk of Nephropathy and risk of Cardiovascular disease were predicted separately.

Subsequently, classification algorithms like Logistic Regression (LR), Support Vector Machines (SVM), Naïve Bayes (NB) and Decision Tree, both with and without AdaBoost, were applied to predict a patient's risk of Nephropathy and risk of Cardiovascular disease. To determine the performance of each algorithm, several metrics like Accuracy Score (ACC), Confusion Matrix, Classification Report, ROC Curve and AUC score were used. At the beginning of the project, the algorithms were applied without using PCA on them, and later PCA was used to find the risk of Nephropathy while both PCA and Oversampling were implemented on the dataset in case of finding the risk of Cardiovascular disease.

4 DATASET

The dataset used in the proposed model was found in a loyalty free dataset sharing platform. It is from an open-label, central registration, multicenter, prospective observational study that was conducted at the Tokyo Women's Medical University Hospital and 69 collaborating institutions in Japan [13]. It has most of the variables needed to implement the proposed Diabetes Complications Prediction Model. The original dataset has 779 instances and 164 variables. Most of these variables are unique and the rest are instances of one variable at different times (e.g. glucose at month 0, glucose at month 3 etc.). In order to make this dataset useful for the project, some unnecessary variables were removed at the beginning of data cleaning phase due to their lack of relation to the project.

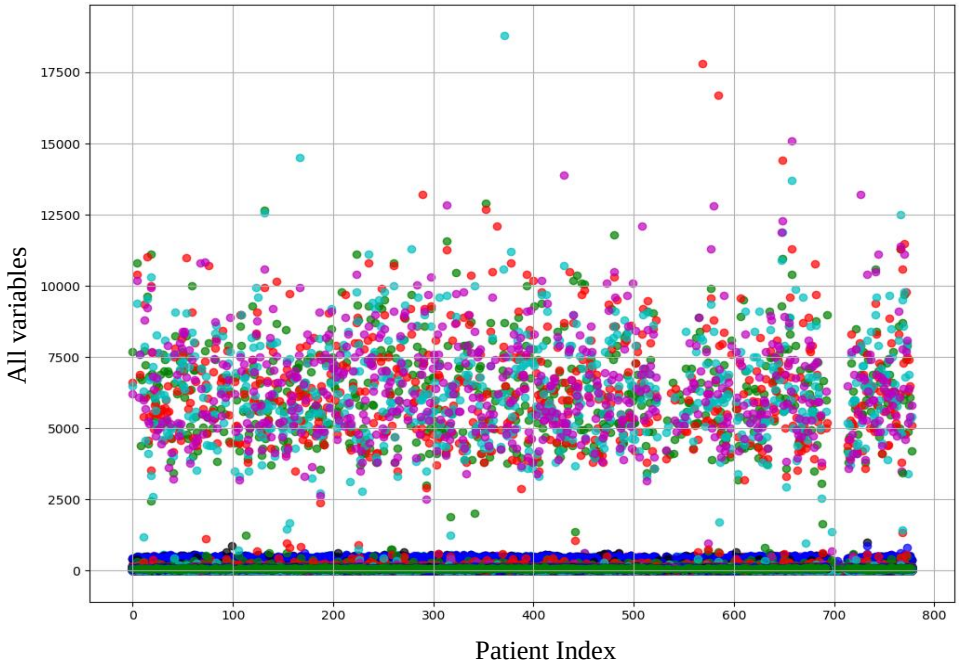


Fig. 2. Dataset Preview.

The above Figure 2 shows the whole dataset in relation with patient index, different colored dots depicts different features; the x-axis represents the number of patients and the y-axis is the values of all the variables, whereas each color portrays each of the 164 features. The cluster of points in the middle illustrates how different variables have values lying within the same range. The whole dataset can be classified into two groups depending on type of variables- numeric and categorical.

4.1 NUMERIC FEATURES

This group of the dataset contains 38 features and most of these features have multiple instances. The features are Age, Height, Weight, Waist circumstances, Duration of T2DM, Blood Pressure (Systolic and Diastolic), Pulse rate, Weight, BMI, HbA1c, Fasting Plasma Glucose, Insulin, HOMA-R, HOMA-beta, C-peptide, CPI, 1,5-anhydroglucitol,1,4 -anhydro (-D)-glucitol, Glycoalbumin, Proinsulin, Na, K, Cl, Triglyceride, HDL-C, LDL-C, BUN, Uric acid, Creatinine, eGFR, Red Blood, White Blood, Hemoglobin, Hematocrit, Platelets, AST, ALT, and γ-GTP. All these numeric features are necessary to determine health condition of a T2DM positive patient. Most of the pathological test results above are related to Nephrology and rest contribute to overall health condition including coronary functions. Height, weight, BMI and waist circumstances determine obesity of a person and obesity is directly linked to Diabetes related complications. Duration of T2DM is an important factor of this project as the duration of Diabetes Mellitus can affect many other factors. Similarly, HbA1c or Glycated hemoglobin test is used to diagnose Diabetes. Regular HbA1c test determines if Diabetes is within range and abnormal results can indicate complications due to Diabetes.

4.2 CATEGORICAL FEATURES

The second group of the dataset, categorical features, contains 21 features and only two of these have multiple instances of over a year. These two variables are Urinary Protein and Urinary Sugar; Urinary Protein is the indicator of amount of protein passing through Kidney. Similar to Creatinine, elevated level of Urinary protein results in Kidney complications. Urinary Sugar test is a method of diagnosing Diabetes similar to HbA1c; it is used to check if medications given to the person to control Diabetes are working properly. Rest of the variables of this group though categorical, contains only binary values of 0 and 1. Hence they have been sub-grouped under binary categorical features.

4.3 BINARY CATEGORICAL FEATURES

The dataset includes 19 binary features - Sex, Smoking habit, Drinking habit, History of complications, History of Hypertension, History of Arteriosclerosis Obliterans, History of Atrial Fibrillation, History of Myocardial Infarction, History of Dyslipidemia, History of Myocardial Infarction, History of Cerebral Infarction, History of Angina Pectoris, History of Heart Failure,

History of Retinopathy, History of Kidney disease, History of Hyperuricemia, History of Liver disease, Risk of Cardiovascular disease and Risk of Nephropathy. All these features contain binary values of either 0 or 1. It is mentionable that some numeric variables change values based on the sex of the instances. History of Smoking and Drinking are taken to consideration due to the fact that both of these affect Heart, Kidney and Liver directly. History of complications indicates if a patient has already suffered from any health complications related to Diabetes. It is a known fact that excessive high blood pressure can result in Hypertension which in turn damages regular Cardiovascular activity to a great extent; History of Hypertension indicates if a patient is suffering from hypertension or not. Dyslipidemia indicates abnormal lipid levels, these lipids can be HDL, LDL or triglyceride. It is another major factor for assessing Heart risks. History of Kidney disease, history of retinopathy and history of Liver disease all respectively indicate if patient already had Kidney, Eye or Liver disease.

4.4 TARGET VARIABLES

4.4.1 Risk of Nephropathy

Nephropathy onset prediction requires an output variable. This variable is necessary to cross examine between the disease onset probability predicted by the proposed system and the real scenario of Nephropathy among the patients. However, the dataset used in the established system did not contain any such variable. Cho (2017) generated a variable using some conditions related to Urinary Albumin (Micro Albumin), History of Renal Failure at the time of Diabetes diagnosis and evidence of diabetic Retinopathy for differential diagnosis of diabetic Nephropathy [16]. Patients are assumed Nephropathy positive if they have Micro Albumin 20—200 mg/min (Microalbuminuria), no evidence of previous Renal failure and previous Retinopathy record. Although, the dataset used in this system has a column for History of Nephropathy, it does not have one for Micro Albumin and which type of Retinopathy record was used by Cho has not been mentioned. Hence, this criteria also became useless. Regardless, it has been found that Microalbuminuria can be determined using albumin/creatinine concentration ratio and a ratio of above 30 confirms Microalbuminuria [17, 18]. Since, record of Retinopathy could not be used, this third criteria has been replaced by eGFR. eGFR less than 60 mL.min-1 determines diabetic Nephropathy [19]. As mentioned earlier, the dataset contains all pathological test results of around a year divided in 4 instances. Hence, for both eGFR and urinary albumin/creatinine ratio,

values of the 12th month has been considered. Finally, the 'risk of Nephropathy' column has been generated as a categorical target where patients are considered positive if either one of the following conditions is true:

1. Urinary albumin/creatinine ratio greater than 30,
2. No history of previous renal complication and
3. eGFR less than 60 mL.min-1.

The risk of Nephropathy variable had 719 instances, out of which 540 belonged to the false class (0) and 179 belonged to the true class (1). The remaining 60 instances from the original dataset were not considered in the model since they already had history of Kidney complications. Only 24.9% patients had risk of Nephropathy.

4.4.2 Risk of Cardiovascular Disease

In order to determine Cardiovascular disease onset among patients an output variable is required. Hence, one of the most important factors in the dataset is the feature risk of Cardiovascular diseases. It is a hybrid column generated from 4 factors, History of Diabetes Mellitus, History of Hypertension, Hypertriglyceridemia and History of Dyslipidemia. The original dataset used in this model did not have any variable that directly indicates patients with current Cardiovascular risks which is vital information for the accuracy of the model. Cercato in his paper described a method on how to directly assess Cardiovascular risks depending upon mentioned 4 variables. The author used Hypercholesterolemia as lipid abnormality indicator whereas for this model History of Dyslipidemia has been used which is a more appropriate lipid abnormality indicator [14]. The dataset used is based on those patients who already have T2DM. Amongst these patients, only those who have Hypertension, Dyslipidemia and Hypertriglyceridemia all have been marked with positive Cardiovascular risk. To make the column more accurate, patients with history of Heart failure also have been marked positive in Cardiovascular risk column. Lastly, in the original dataset there was no such column as Hypertriglyceridemia, this column has been generated from the triglyceride variables [15]. Only those patients who show higher than normal (150 mg/dL) amount of Triglyceride have been marked as Hypertriglyceridemia positive. The Cardiovascular risk variable had 778 instances, out of which 703 belonged to the false class (0) and 75 belonged to the true class (1) with 1 instance being removed, since there were not sufficient data on the patient to determine whether he/she had any Cardiovascular risk. Only 9.64% of the patients had Cardiovascular risk.

5 SYSTEM IMPLEMENTATION

5.1 MISSING VALUE IMPUTATION

The dataset had values missing at random (MAR). Missing at random means the tendency of an instance to be missing is related to the observed data and not the missing data. Hence, the missing value of a variable can be predicted from the other variables [20]. For example, missing HbA1c values maybe lower or higher than what it would have been if actually measured because HbA1c depends on Age, Duration of T2DM, Fasting Plasma Glucose and some other values as well.

Significant amount of information is discarded due to the presence of missing values in a dataset. Therefore, imputation of missing data through the proper algorithms are needed in order to get better insights about the data. Techniques like removal of entire instances, mean imputation and replacing values with the value zero are some simple methods of imputations. However, all of these methods are inefficient in this case. In this project, only one instance was entirely deleted from the dataset since it had missing values for most of the variables. Afterwards, algorithms like Random Forest and missForest were applied to impute the data.

5.1.1 Missing Data Visualization

Visualization of incomplete data allows to simultaneously explore the data and the structure of missing values. This is helpful for learning about the distribution of the incomplete information in the data, and to identify possible structures of the missing values and their relation to the available information.

Before data imputation, it is required to analyze the dataset including the missing data to acquire knowledge on the completeness and authenticity of the dataset. Using the package “missingno” in Python, the below graph was generated to analyze the dataset better.

The nullity matrix in Figure 3 is a data-dense representation of all the 164 variables and their consistency across the 778 instances. The graph helps in visually picking out the pattern in the data consistency quickly. The Sparkline at the right summarizes the general shape of the data completeness including the maximum and minimum rows. Through this graph, it is evident that few variables have more than 50% missing data, which may impact the data imputation and consequently affect the final outcome.

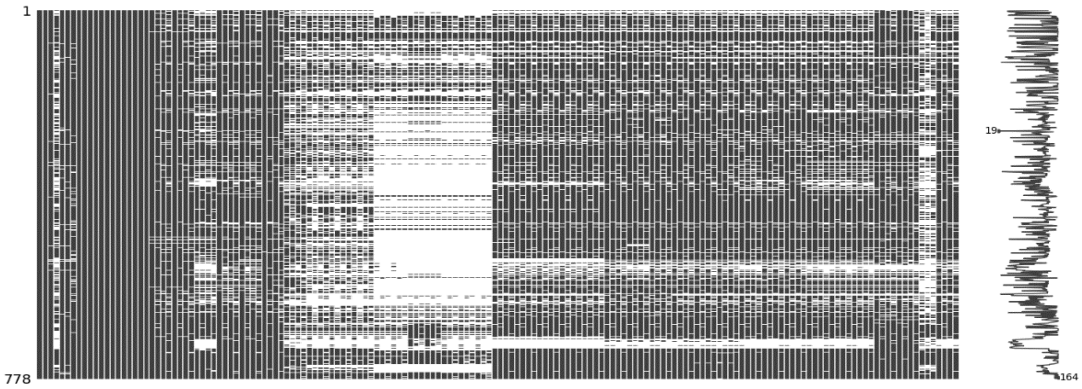


Fig. 3. Nullity Matrix representing missing data.

5.1.2 Random Forest Imputation

Random Forest is a supervised learning algorithm used for classification and regression. It consists of several Decision Trees which makes a prediction on its own. Then by aggregating the predictions of each of the trees, the new value is estimated [21]. The idea was used to make a Random Forest regression model that predicts the value of each missing instance of every variable by comparing it with the rest of the variables. It is possible to obtain a good balance between AUC, processing time, and memory usage when the number of trees in a forest is between 64 and 128 [22]. Hence, in this model, the forest consisted of 64 trees.

For each variable in the data it was regressed with the rest of the data, so for variable v_1, it was regressed with $v_{2 \ldots} v_n$, which do not have an overlap in missing data with v_1. The indexes of the instances that have missing data for v_1 were added to a list and the overlap of missing values between variable one and the other variables were determined. Then only the instances that do not have overlapping missing data with v_1 were used to fit the Random Forest model. After that, v_2 was regressed with v_1, v_3... v_n, and so on. This way a Random Forest regression model based

on non-missing data was made. After fitting the regression, the predictors (v_2... v_n) were used to predict the missing data in v_1… v_n and imputed with that value.

This model does not handle categorical variables but instead treats them as numeric ones. Since the dataset consists of both categorical and numeric variables, this is a problem. Furthermore, Random Forest is also computationally expensive since the processing time is very high. Figure 4 shows the data-points before and after imputation using Random Forest of some selected variables. The blue dots represent the actual data and the red dots represent the imputed data.

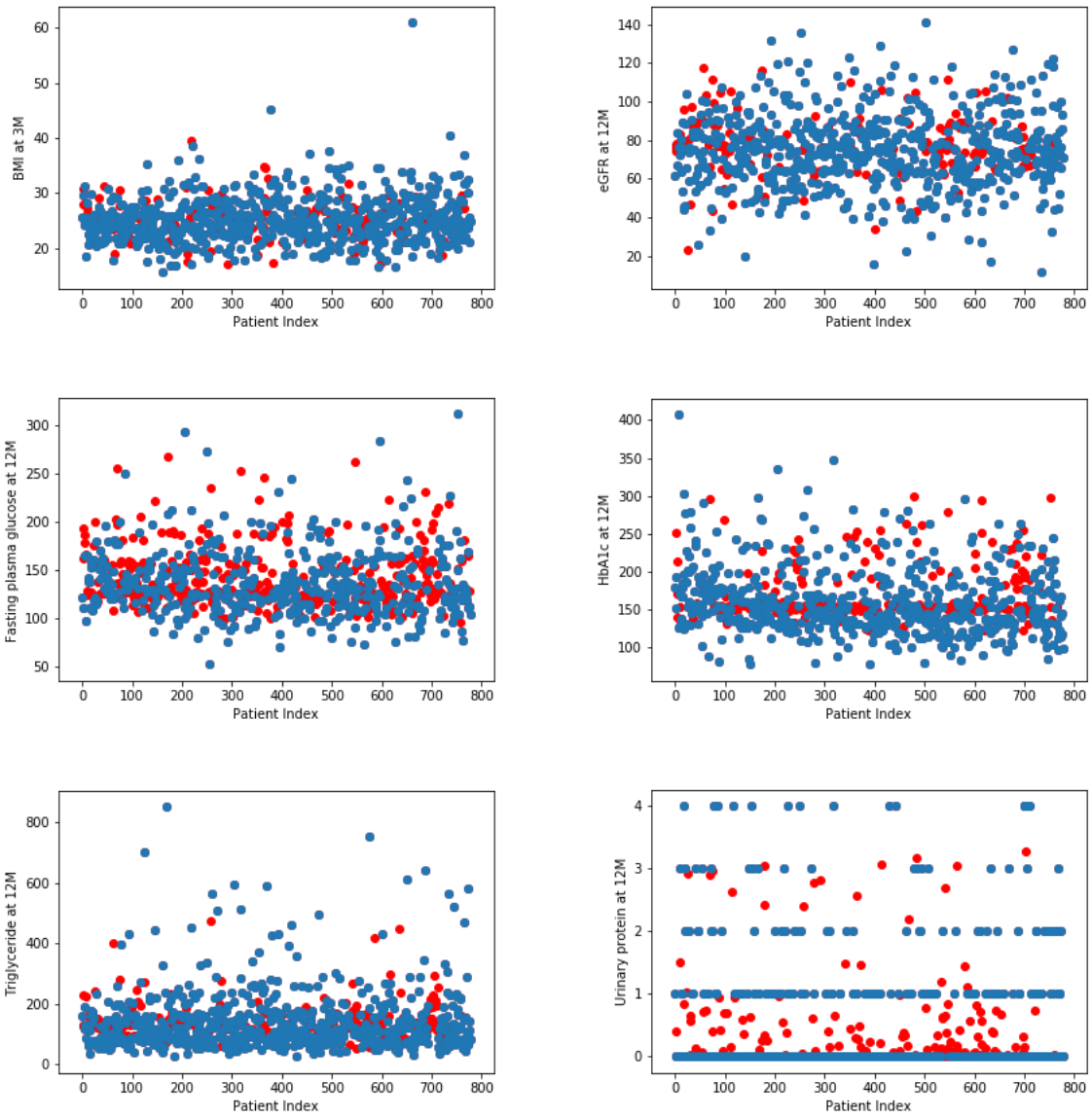


Fig. 4. Few variables after Random Forest Imputation.

5.1.3 missForest Imputation

Most imputation algorithms cannot handle mixed-type data that contains both numeric and categorical variables. However, the missForest algorithm is an exception. It uses a non-parametric method to impute missing values by not taking the data type into consideration [23]. Figure 5 shows the data-points before and after imputation using missForest of some selected variables. The blue dots represent the actual data and the red dots represent the imputed data.

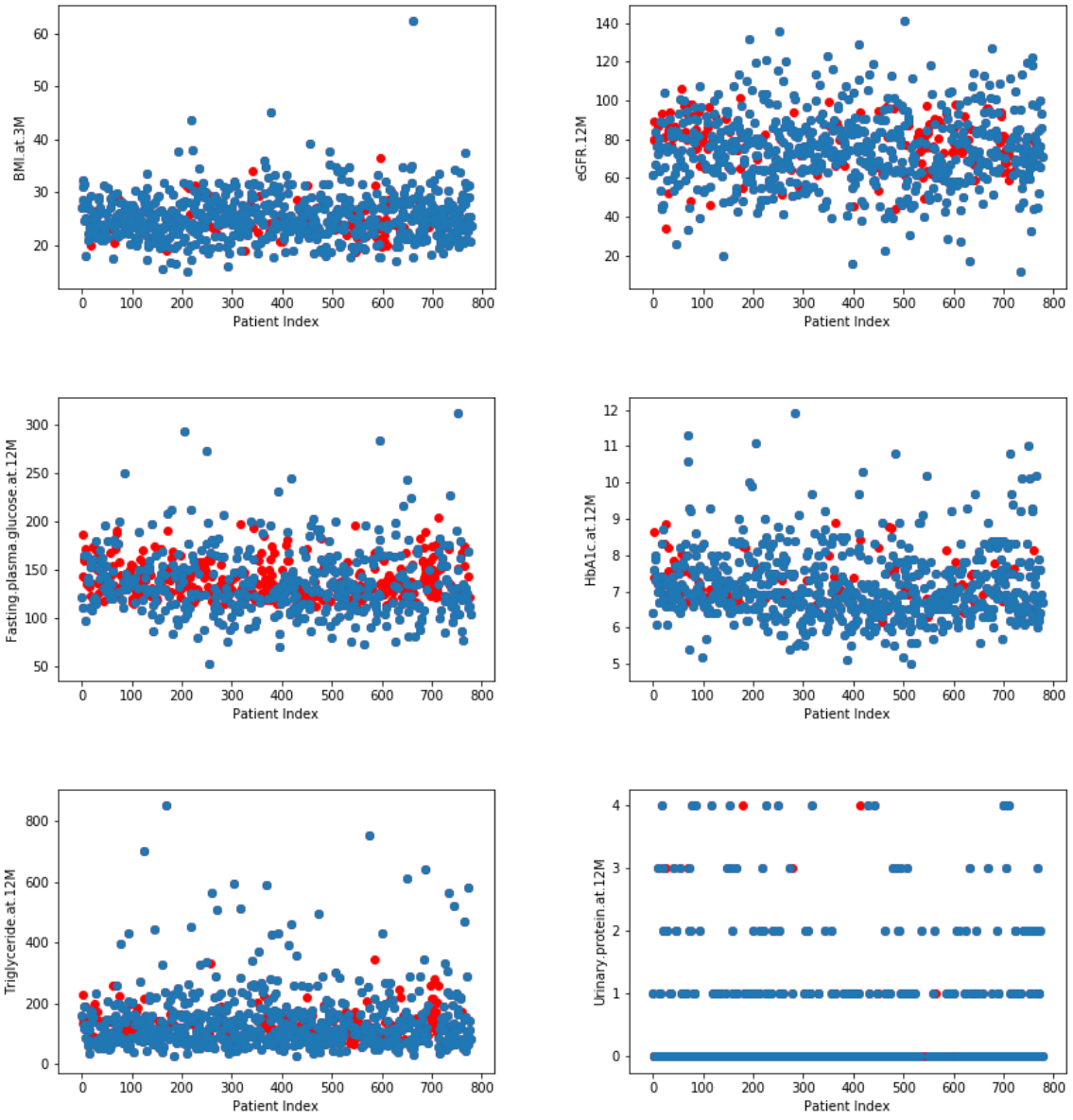


Fig. 5. Few variables after missForest Imputation.

It was implemented using R's missForest package, which was run inside a python package called rpy2. Rpy2 acts as a python interface to the R language. In each forest of the algorithm 100 trees were used and the maximum number of iterations to be performed was 10. However, the missing values converged with only 7 iterations and took considerably less processing time than Random Forest imputation.

The model yields an out-of-bag (OOB) imputation error estimate which consists of the NRMSE (Normalized Root Mean Squared Error) for the continuous numeric variables and PFC (Proportion of Falsely Classified) for categorical variables. NRMSE for this model, the NRMSE was 17.89% and the PFC was 12.19%. This means the continuous variables were imputed with an error of 17.89% while the categorical variables were imputed with an error of 12.19%.

5.2 CATEGORICAL VARIABLE CONVERSION

The dataset consisted of several categorical variables: Gender, History of Complications, History of Kidney Disease, Urinary Sugar, Urinary Protein, etc. However, since these variables have numerical values, algorithms will treat them like numerical variables. For example, for the variable gender, 1 represents a male patient while 2 represents a female patient. So any classification or dimensionality reduction algorithm will give much more importance to a female patient since it has a higher numerical value.

To counter this problem, all the categorical variables was converted to its numerical counterparts. This was done by creating dummy variables for each class present in every categorical variable. So, the variable gender, had one column representing male patients with a value of either 0 or 1 depending on whether the patient is a male or not. And similarly, one column representing female patients.

This was implemented in python using the OneHotEncoder class of the Scikit-Learn library [24].

5.3 FEATURE SCALING

Feature scaling is a method to scale data of different independent variables with varying ranges to a standardized range which is consistent across all the features. There are various types of feature scaling techniques that can be applied to standardize the range. The technique being applied in this project is standardization; standardization replaces the values of a particular

feature with their Z scores. This redistributes the features with their mean μ = 0 and standard deviation σ =1. The scikit-learn module sklearn.preprocessing.scale is used to implement standardization in python. Below is the formula of standardization:

$$x' = \frac{x - \bar{x}}{\sigma} \quad (1)$$

Corresponding to each variable, x is the original feature vector, $\bar{x}$ is the mean of that feature vector and σ is its standard deviation. Subtracting the mean from the original value and dividing it by the standard deviation leads to the scaled value of the data. Standardization is particularly important since some classification algorithms like SVM, Logistic Regression and Neural Networks do not perform well on unscaled data. Furthermore, data should be standardized before applying PCA as variables with higher and lower variance are going to be treated differently. Hence, a variable with a large scale would always be considered the first principal component.

5.4 PRINCIPAL COMPONENT ANALYSIS (PCA)

PCA or Principal Component Analysis is a dimensionality reduction technique that uses orthogonal transformation to reduce a large set of variables to a smaller set of variables while retaining most of the information present. The procedure works by converting the highly correlated variables of the original dataset to a smaller number of uncorrelated linear variables called principal components. These principal components then account for most of the variance of the original dataset [25].

PCA is very useful when the data has dimensions of 3 or higher since it becomes extremely hard to make predictions from such huge amount of information. Furthermore, visualization of high dimensional data is also difficult. PCA solves this problem regarding data visualization as well.

Number of principal components is less than or equal to the smaller of the number of original features or the number of observations. Initially, the dataset comprised 779 observations and 164 useful features, so the maximum number of principal components was 164.

Out of those 164, 50 components were taken which attributed to 80% of the original variance as shown in Figure 6.

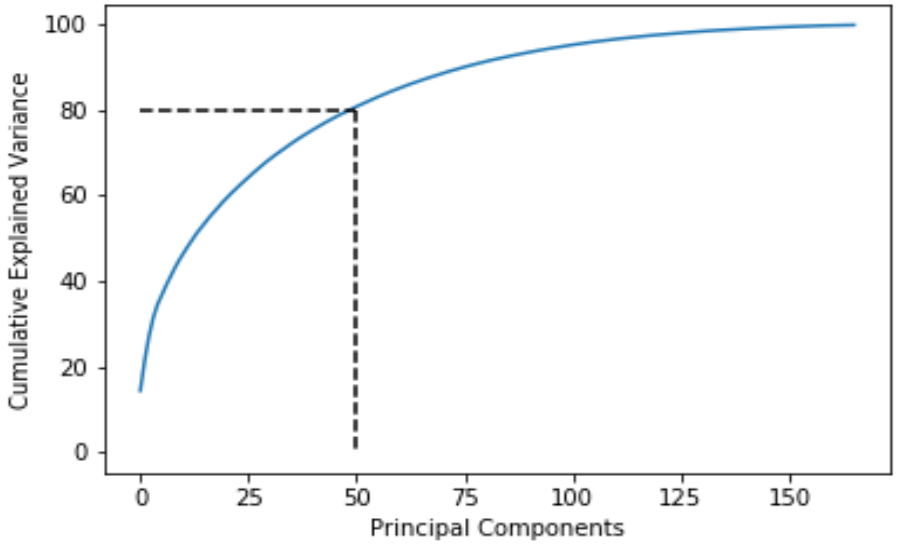


Fig. 6. Cumulative Explained Variance against No. of Principal Components.

After applying PCA the dataset is reduced to 50 principal components that represent each instance. Figure 7 shows the dataset after PCA.

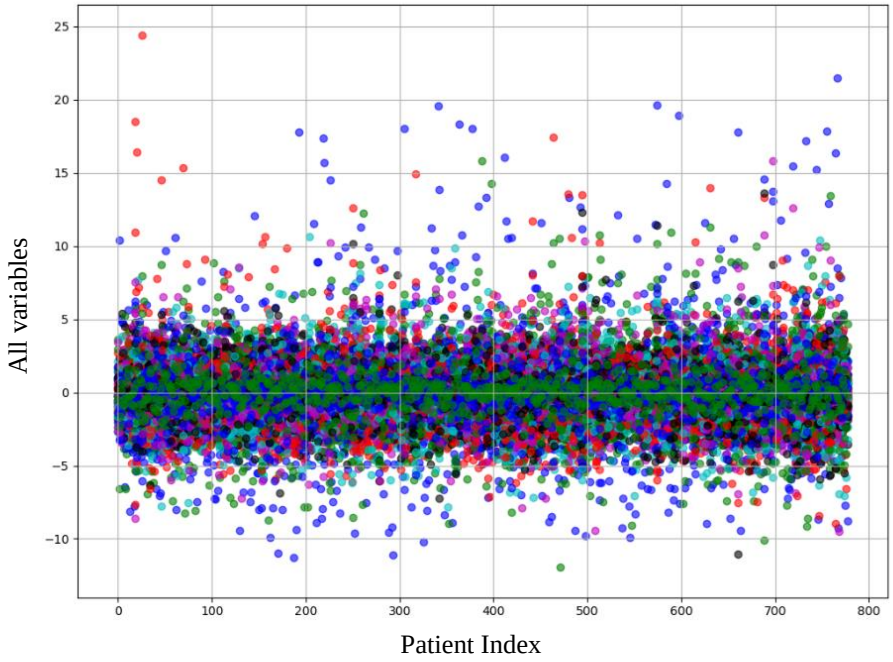


Fig. 7. Dataset after applying PCA.

5.5 CLASS IMBALANCE AND OVERSAMPLING

If a dataset has significantly more instances of one class and fewer instances of the other class, then the dataset is considered being imbalanced [26]. This dataset of diabetic patients includes fewer instances where the disease was present (minority class) and more instances where it was not present (majority class). In the dataset, the number of patients diagnosed with Nephropathy is 179, compared to the number of patients without Nephropathy being 541. On the other hand, only 75 patients were diagnosed with Cardiovascular disease while the rest 703 patients did not have any complication. This shows how unbalanced and biased the dataset is. If the dataset is left as it is, the minority class tends to achieve poor predictive accuracy as the algorithm predicts the majority class more often [27]. To counter this problem, the classes are balanced by either increasing the minority class to equal the majority class or to decrease the majority class to equal the minority class. Instead of decreasing the majority class, the minority class was increased, since the dataset contained only few instances, which is also known as Oversampling.

The first step in oversampling was dividing the dataset into training and test set. After that, the training set was oversampled using the SMOTE algorithm (Synthetic Minority Oversampling Technique). Oversampling both the training and test set instead of only the training set may lead to information loss in the training set. This causes the training set to have a bad Recall score. SMOTE creates synthetic instances of the minority class by finding similar instances using k-nearest-neighbors. Then it randomly chooses one of those neighbors and modifies it to create a new instance of the minority class [28]. It was implemented using the SMOTE class of the imbalanced-learn package of python. A class ratio of 1 was used in order to equal the minority and majority class examples.

5.6 TRAIN-TEST SPLIT

When it comes to Machine Learning models, a common problem faced is with overfitting and underfitting. Overfitting occurs when the model learns from the data so well that it leads to no training error. It learns the noises and deviations of the data as concepts and fits the model too well to the data. However, when the model is presented with new data, it fails to make accurate predictions since it was over fit with the noises and deviations [29]. Underfitting is the opposite, it can neither fit to the data available nor to new data.

The remedy to these problems is cross-validation. This is the process of splitting the data into two sets; the training set, used to train the data and the test set, used to test the data. By introducing the data to the test set which is new data, it can be determined if the model is overfitting.

The train/test split was implemented using the train_test_split class of scikit-learn's model_selection package. The data was split into a 70:30 ratio with 70% going to the training set and the rest to the test set. In most cases this is the ideal ratio of splitting data [30]. A 50:50 ratio was also considered but since the dataset did not contain a large number of instances, taking only 50% of the original data might have led to underfitting of the model.

The initial dataset is imbalanced. A dataset is considered to be imbalanced if it has significantly more instances of one class and fewer instances of the other class as stated by Perveen et al [26]. This dataset of diabetic patients includes fewer instances where the disease was present (minority class) and more instances where it was not present (majority class). For example, the number of patients who were not diagnosed with Kidney complications were 541 compared to the 179 who were. Similarly, only were diagnosed with Cardiovascular complications while were not.

For Nephropathy, the test set consisted of 216 instances. Out of these 216 instances, 162 belonged to the false class (does not have Kidney complications) and 54 belonged to the true class (has Kidney complications). The parameter stratify is used to make sure the distribution of classes of in the test set is the same as that of the original dataset. In the case of Cardiovascular disease, the test set consisted of 234 instances. Out of which, 211 instances belonged to the false class (does not have Cardiovascular complications) and 23 belonged to the true class (has Cardiovascular complications). The parameter stratify is used to make sure the distribution of classes in the test set is the same as that of the original dataset. When oversampling was applied in the Cardiovascular disease prediction model, 10-fold cross validation was used instead of train-test split. Since only the training set was oversampled, 10-fold cross validation seemed a better option for enhanced performance.

5.7 ALGORITHMS

The problem at hand is a multi-label classification problem, which is a form of classification. Unlike binary or multi-class classification, an instance can belong to either or both of the output classes. For example, in this paper, a patient can have either Kidney complications, Heart complications or both at the same time. The best way to solve this issue, is by problem transformation, where a multi-label problem is transformed into a single-label binary problem [31]. So for a particular patient, it is determined separately whether he/she will have Kidney or Heart complications using classifier algorithms for single-label problems. Afterwards, the results are then combined to produce multi-label predictions for each patient.

The single-label classifiers used are Logistic Regression (LR), Support Vector Machines (SVM), Decision Tree (DT), AdaBoost, Random Forest (RF) and Naïve Bayes (NB). The results of the six algorithms were compared to determine the best classifier for the problem.

5.7.1 Logistic Regression (LR)

Logistic Regression is a supervised learning algorithm that trains the model by taking input variables(x) and a target variable(y). In Logistic Regression the output or target variable is a categorical variable, unlike Linear Regression, and is thus a binary classification algorithm that categorizes a data point to one of the classes of the data [32]. The general equation of Logistic Regression is:

$$\boldsymbol{log}\left(\frac{\boldsymbol{p(X)}}{\boldsymbol{1-p(X)}}\right) = \boldsymbol{\beta_0} + \boldsymbol{\beta_1 X} \qquad (2)$$

Where, p(X) is the dependent variable, X is the independent variable, β_0 is the intercept and β_1 is the slope co-efficient.

Logistic Regression measures the relationship between the dependent variable, the output, and the independent variables, the input, by estimating probabilities using its underlying logistic function. It uses L2 penalty for regularization. The resultant probabilities are then converted to binary values 0 or 1 by the logistic function, also known as the sigmoid function. The sigmoid function takes any real-valued number and maps it into a value between the ranges 0-1 excluding the limits themselves. Afterwards, a threshold classifier transforms the result to a binary value.

One of the primary assumptions of Logistic Regression is that the input features should be independent of each other, i.e. one variable should have little or no co-linearity with the other variables [33]. Hence, PCA is executed on the data beforehand, to convert the correlated variables to a set of uncorrelated variables.

5.7.2 Support Vector Machines (SVM)

Support Vector Machines, SVM, is a supervised learning model with associated learning algorithms that analyze data used for classification, regression analysis and outlier detection [34, 35]. It is a non-probabilistic binary linear classifier, but can be manipulated in a way that it can perform non-linear and probabilistic classification as well, making it a versatile algorithm. An SVM model is a representation of the instances as points in space mapped so that they can be categorized and divided by a clear gap. New instances are then mapped into the same space and predicted in which category it might be in based on which side of the gap they fall in. The main advantage of SVM is the fact that it is effective in high dimensional spaces. Additionally, it is also memory efficient since it uses a subset of training points in the decision function.

The SVM in scikit-learn supports both dense and sparse sample vectors as input, even though optimal performance is usually obtained while using dense datasets. In this project, SVC class from scikit-learn was used to implement SVM.

5.7.3 Naïve Bayes (NB)

Naïve Bayes is a supervised learning algorithm that depends on Bayes' Theorem for classification. Bayes' theorem uses conditional probability which in turn uses prior knowledge to calculate the probability, that a future event will take place. The formula for Bayes' Theorem is:

$$\boldsymbol{P(H|E)} = \frac{\boldsymbol{P(E|H)} * \boldsymbol{P(H)}}{\boldsymbol{P(E)}} \tag{3}$$

Here, P(H|E) is the posterior probability, the probability that a hypothesis (H) is true given some evidence (E). P(H) is the prior probability, i.e., the probability of the hypothesis being true. P(E) is the probability of the evidence, irrespective of the hypothesis. P(E|H) is the probability of the evidence when hypothesis is true.

In Naïve Bayes classifier, it is assumed that the input variables (features) are independent of each other and that all features will individually contribute to the probability of the target variable. So,

the existence of one feature variable does not affect the other feature variables. This is why it is called naïve. However, in real datasets, the feature variables are usually dependent on each other so this is one of the drawbacks of the Naïve Bayes classifier. Naïve Bayes classifier though, works very well for large data sets and sometimes performs better than other complicated classifiers. There are several different types of Naïve Bayes classifiers, among them, the Gaussian Naïve Bayes classifier was used in this model. The Gaussian Naïve Bayes classifier assumes that the feature values are continuous and the values of belonging to each class are normally distributed [36].

Naïve Bayes algorithm is used for binary and multiclass classification and can also be trained on a small dataset which is a huge advantage. It is also very fast and scalable. Moreover, it mitigates the problem arising from the curse of dimensionality to some degree. However, as mentioned before, it makes the unrealistic assumption that the input variables are independent of each other. This is not the case in real life datasets, where there can be many complex relationships between the feature variables.

5.7.4 Decision Tree (DT)

Decision Tree is a supervised learning algorithm that is used for classification and regression. It works by splitting the data into two or more subsets based on the values of the input variables [37, 38]. A cost function or splitting criterion is used to determine the best split (one with the lowest cost) among all the split points. The data is split recursively into groups until the leaves contain only one sample. In this model, an optimized version of the CART (Classification and Regression Trees) algorithm is used to implement the Decision Tree classifier using Scikit-Learn. Gini impurity is used as the splitting criterion to measure the uncertainty. Decision Trees are easy to interpret and understand, compared to other classification algorithms. Moreover, Decision Trees require little preprocessing as outliers do not affect the performance. Furthermore, they are not based on the Euclidean distance, hence, feature scaling is not required. Also, feature scaling could lead to wrong assumptions being implied since the values would be changed. Decision Trees can handle both categorical and numerical variables as input so it is appropriate for this model, since the dataset contains both variable types. In this model, the relationship between the feature variable and target variable is complex and highly non-linear. So a Decision Tree has a greater chance of outperforming linear models like Logistic Regression. Even though

Decision Trees have several advantages, they also have a few disadvantages. One is that, Decision Trees can cause overfitting by making a tree that is too complex and hence does not predict well on new data. Finally, since Decision Trees are greedy algorithms, the optimal tree is not necessarily returned.

5.7.5 AdaBoost

AdaBoost or Adaptive Boosting is a boosting technique that is used with Decision Trees in order to elevate their performance [39]. A disadvantage of Decision Trees is, any small change in the data can cause a completely different tree to be produced. This is called variance and can be tackled using boosting techniques like AdaBoost. It works by first fitting the Decision Tree classifier on the dataset and then fitting extra samples of the same classifier on the same dataset. However, the extra samples are fitted by changing the weights of the instances that were falsely classified so that the following classifiers put more emphasis on bad cases. The equation for classification can be represented by:

$$\boldsymbol{F(x) = sin(\sum_{m=1}^{M} \theta_m f_m(x))} \tag{4}$$

Here, f_m stands for the m^{th} weak classifier and θ_m is the corresponding weight. So it is the weighted combination of M weak classifiers that are turned into a strong classifier. AdaBoost is implemented using sklearn's ensemble.AdaBoostClassifier class, which uses the AdaBoost-SAMME algorithm [40].

5.7.6 Random Forest (RF)

The basic concept of the Random Forest algorithm has already been discussed in section 7.2. It was used to predict missing values by modelling the Random Forest algorithm as a regression problem. However, in this case, it is modelled as classification problem since Random Forest works for both. Since Decision Tree learners can overfit the data and also produce trees that are biased towards one class. Random Forest is used to increase the performance by avoiding overfitting and bias through aggregation of several trees. The best performance was achieved with a Random Forest consisting of 80 trees. Just like with Decision Trees, Gini impurity was used as the underlying criterion for splitting the data.

6 RESULT ANALYSIS

After implementation of different Machine Learning models, the next step is to find out how the models performed. This is done by running the models on the test dataset which was set aside earlier. The test dataset comprised of 30% of the original data for Nephropathy. 10-fold cross-validation was done for Cardiovascular disease using 10% of the original data in the test dataset. To determine and compare the performance of the different algorithms, several performance metrics were used.

6.1 PERFORMANCE METRICS

The performance of Machine Learning algorithms is evaluated using several performance metrics. Performance metrics relating to classifications are discussed here since the paper only deals with classification problems. For Nephropathy, if the target variable (risk of Nephropathy) is 1 then it is a positive instance, meaning the patient has Kidney complications. And if the target variable is 0, then it a negative instance, meaning the patient does not have Kidney complications. Similarly, for Cardiovascular disease, if the target variable (risk of Cardiovascular disease) is 1 then it is a positive instance, meaning the patient has Heart complications. And if the target variable is 0, then it a negative instance, meaning the patient does not have Heart complications.

6.1.1 Confusion Matrix

Confusion Matrix is the easiest way to determine the performance of a classification model by comparing how many positive instances were correctly/incorrectly classified and how many negative instances were correctly/incorrectly classified. In a Confusion Matrix, the rows represent the actual labels and the columns represent the predicted labels. Table 1 shows the Confusion Matrix.

Table 1. Confusion Matrix

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

True Positives (TP): True positives are the instances where both the predicted class and actual class is True (1), i.e., when a patient actually has complications and is also classified by the model to have complications.

True Negatives (TN): True negatives are the instances where both the predicted class and actual class is False (0), i.e., when a patient does not have complications and is also classified by the model as not having complications.

False Negatives (FN): False negatives are the instances where the predicted class is False (0) but actual class is True (1), i.e., when a patient is classified by the model as not having complications even though in reality, they do.

False Positives (FP): False positives are the instances where the predicted class is True (1) while actual class is False (0), i.e., when a patient is classified by the model as having complications even though in reality, they do not.

6.1.2 Accuracy (ACC)

Accuracy determines the number of correct predictions over the total number of predictions made by the model. Even though it is widely used, it is not a very good measure of performance especially when the dataset is imbalanced like in this case. The formula for Accuracy is:

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP} + \boldsymbol{TN}}{\boldsymbol{TP} + \boldsymbol{FP} + \boldsymbol{TN} + \boldsymbol{FN}} \tag{5}$$

6.1.3 Precision

Precision is a measure of the proportion of patients that actually had complications among those classified to have complications by the system. The formula for Precision is:

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FP}} \tag{6}$$

6.1.4 Recall / Sensitivity

Recall or sensitivity is a measure of the proportion of patients that were predicted to have complications among those patients that actually had the complications. The formula is:

$$\boldsymbol{Recall = \frac{TP}{TP + FN}} \tag{7}$$

6.1.5 Specificity

Specificity is the opposite of Recall. It is a measure of the number of patients who were classified as not having complications among those who actually did not have the complications. The formula is:

$$\boldsymbol{Specificity = \frac{TN}{TN + FP}} \tag{8}$$

6.1.6 F1 Score

F1 Score is the harmonic mean of the Recall and Precision that is used to test for Accuracy. The formula is:

$$\boldsymbol{F1 = 2 * \frac{Precision * Recall}{Precision + Recall}} \tag{9}$$

6.1.7 ROC Curve

ROC or Receiver Operating Characteristics is a graphical plot of sensitivity against (1-Specificity) or in other words, a comparison of true positive rate (TPR) and false positive rate (FPR). It is used to visualize a classifier's performance at different thresholds to determine the best threshold point for the classifier [13]. Figure 8 represents an example of ROC Curve.

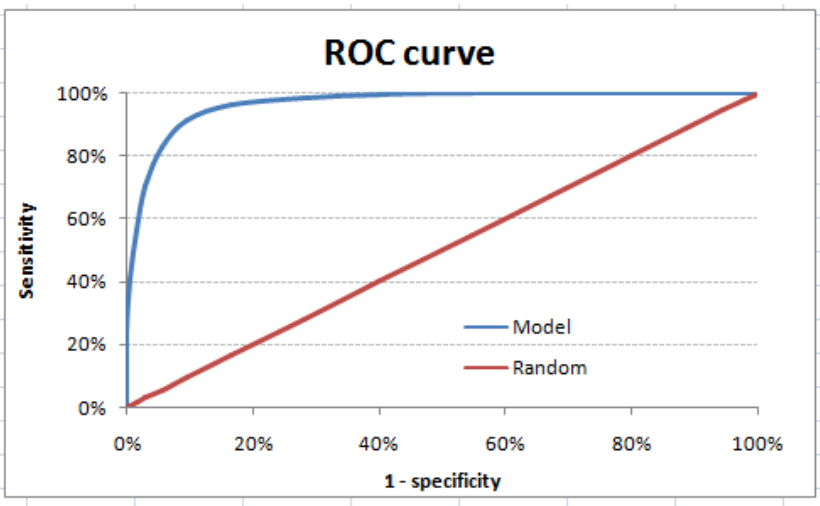


Fig. 8. Receiver Operating Characteristics (ROC) Curve.

6.1.8 Area under the ROC Curve (AUC)

AUC is the entire area under the ROC Curve that is used to determine the performance of a classifier across all classes. It ranges from 0 to 1, higher the value, better the performance. Figure 9 is the example of graphical representation of an AUC Curve. The x-axis represents the false positive rate, or 1-Specificity, and the y-axis represents the true positive rate, or Sensitivity. The shaded region below the blue curve is the AUC in this graph.

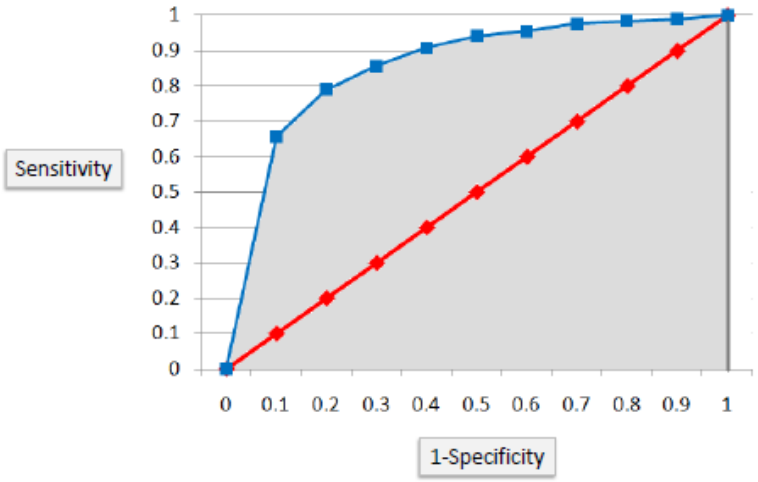


Fig. 9. Area under the ROC Curve (AUC).

6.2 MODEL PERFORMANCES

6.2.1 Nephropathy

In the model for Nephropathy, the set of six classification algorithms was used- Logistic Regression (LR), Support Vector Machines (SVM), Naïve Bayes (NB), Decision Tree (DT) with and without AdaBoost and Random Forest (RF) were applied on the dataset. For Table 2, the algorithms were implemented directly, while in Table 3, the performance metrics are obtained when the same set of algorithms were applied after PCA. In each experiment, the performance was measured using Accuracy, Precision, Recall, Specificity, F1 Score, ROC Curve and AUC Score. The subsequent tables contain all the performance metrics for all the experiments done in the model to predict risk of Nephropathy.

The table below illustrates the results of different performance metrics for the algorithms to detect risk of Nephropathy without PCA.

Table 2. Nephropathy Scores without PCA

	Logistic Regression	SVM	Naive Bayes	Decision Tree	Decision Tree AdaBoost	Random Forest
Accuracy	0.79	0.82	0.79	0.89	0.87	0.89
Precision	0.57	0.86	0.70	0.79	0.71	0.94
Recall	0.67	0.35	0.30	0.81	0.81	0.63
F1 Score	0.62	0.50	0.42	0.80	0.76	0.76
Specificity	0.83	0.98	0.96	0.93	0.88	0.98
AUC Score	0.75	0.67	0.63	0.87	0.85	0.81

The table below illustrates the results of different performance metrics for the algorithms to detect risk of Nephropathy with PCA.

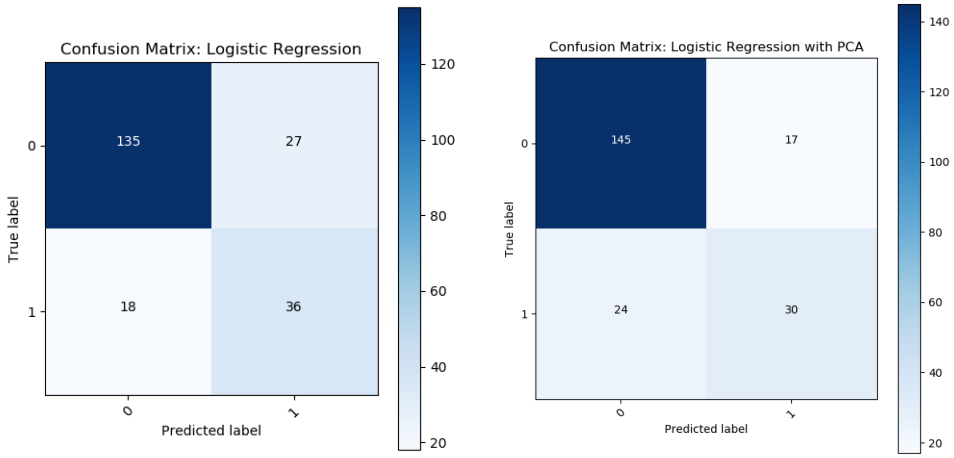
Table 3. Nephropathy Scores with PCA

	Logistic Regression	SVM	Naive Bayes	Decision Tree	Decision Tree AdaBoost	Random Forest
Accuracy	0.81	0.76	0.76	0.76	0.76	0.79
Precision	0.64	1.00	0.51	0.53	0.52	0.68
Recall	0.56	0.04	0.41	0.52	0.50	0.31
F1 Score	0.59	0.07	0.45	0.52	0.51	0.43
Specificity	0.89	0.0	0.87	0.85	0.85	0.95
AUC Score	0.73	0.52	0.64	0.68	0.67	0.63

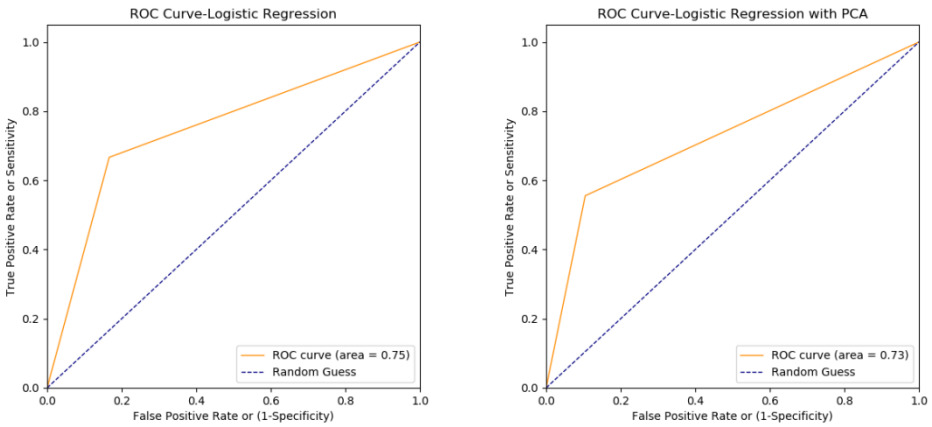
In the following sub-sections, the Confusion Matrix and ROC Curve for every experiment is represented through figures. As shown in Table 1, the Confusion Matrix has four values- True Negative, False Positive, False Negative and True Positive. In all the figures of Confusion Matrix, the blocks represent correctly predicted negative in True Negative, falsely predicted positive in False Positive, wrong prediction of negative in False Negative and correctly predicted positive in True Positive respectively. These values are later used to find the Accuracy, Precision, Recall, Specificity and F1 Score to evaluate the performance of each algorithm. In addition to Confusion Matrix, an ROC Curve which represents the AUC of the algorithms is shown.

6.2.1.1 Logistic Regression

Figure 10(a) illustrates the Confusion Matrix of the Logistic Regression model. Figure 10(b) illustrates the Confusion Matrix of the Logistic Regression model after PCA is applied on the dataset. Figure 10(c) illustrates the ROC Curve of the Logistic Regression model while Figure 10(d) shows the ROC Curve of the Logistic Regression model with PCA. The results show how Logistic Regression performs well for this problem with an accuracy of around 0.80 and AUC score of around 0.75. Introduction of PCA leads to an increase in accuracy and precision. However, it leads to a fall in recall and F1 Score. Since recall is more important than precision in disease detection, Logistic Regression without PCA is the better model among the two.



(a) (b)

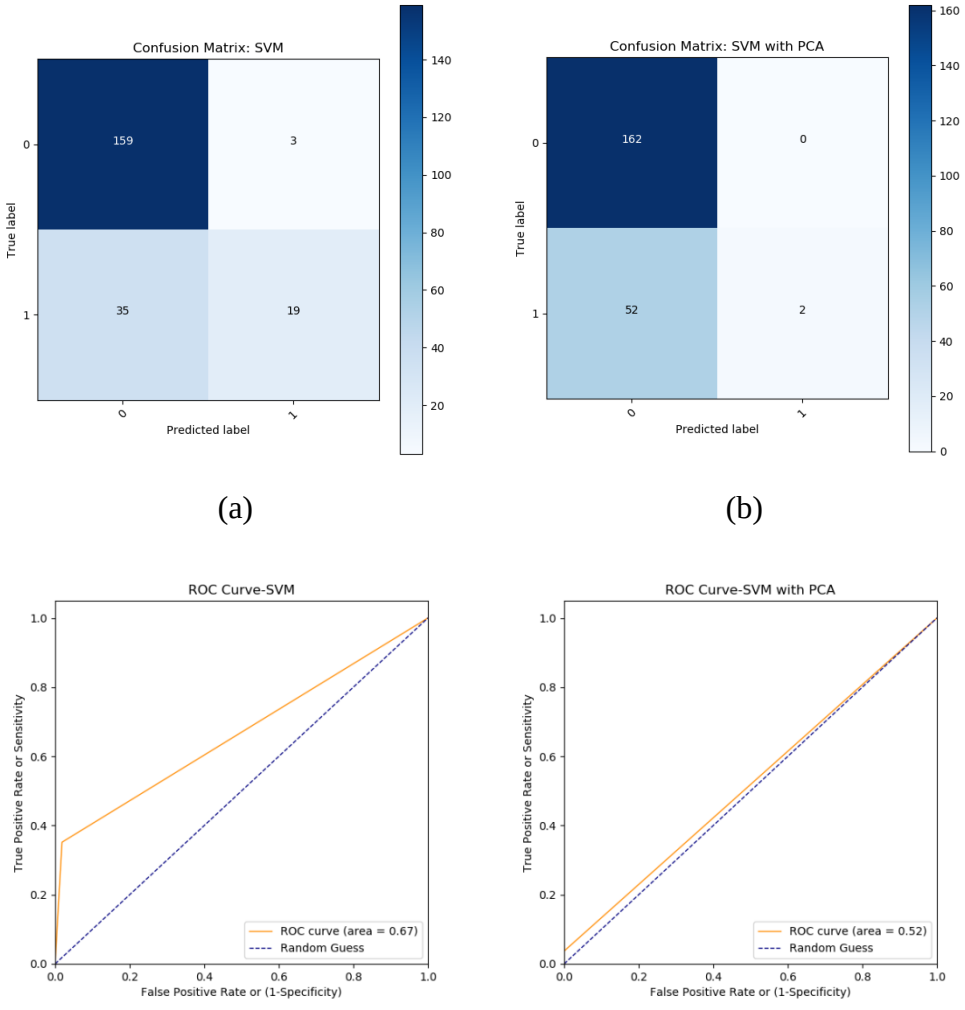


(c) (d)

Fig. 10(a). Confusion Matrix of LR, 10(b). Confusion Matrix of LR with PCA, 10(c). ROC Curve of LR and 10(d). ROC Curve of LR with PCA.

6.2.1.2 Support Vector Machines (SVM)

Figure 11(a) and Figure 11(b) illustrate the Confusion Matrix of the SVM model without and with PCA respectively. Figure 11(c) illustrates the ROC Curve of the SVM model while Figure 11(d) shows the ROC Curve of the SVM model with PCA. In both cases it can be seen from the results that SVM has good accuracy of around 0.80 but the AUC score is poor. The precision score is very high; however, the recall score is poor in both cases. Introduction of PCA leads to the recall falling to 0.04 from 0.35 and F1 Score falling from 0.50 to 0.07. The AUC score also drops by 0.15. This shows that introduction of PCA significantly decreases the performance of the SVM model so the model is better without PCA.



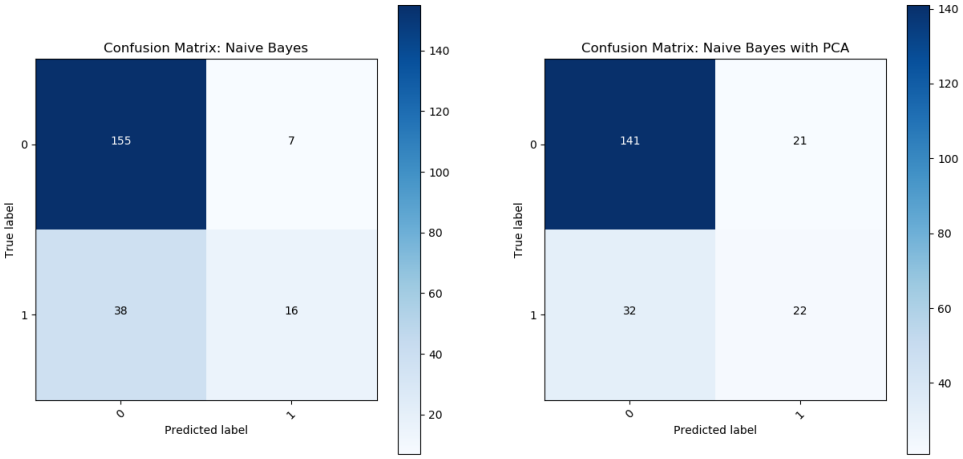
(a) (b)

(c) (d)

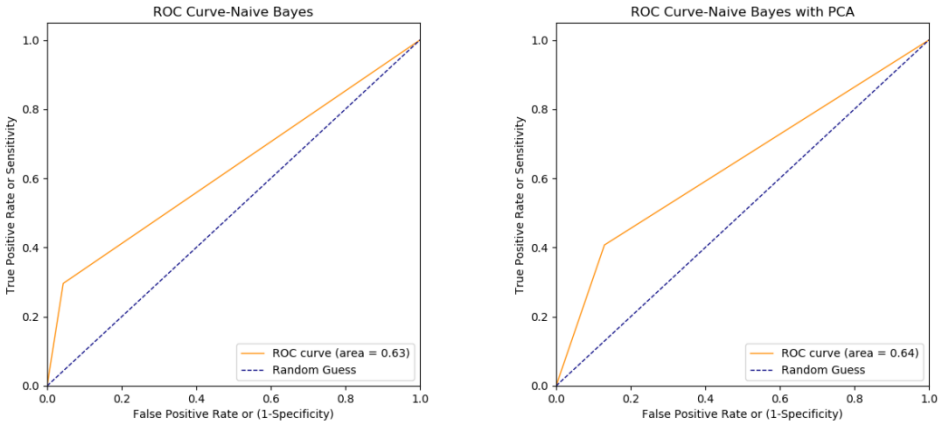
Fig 11(a). Confusion Matrix of SVM, 11(b). Confusion Matrix of SVM with PCA, 11(c). ROC Curve of SVM and 11(d). ROC Curve of SVM with PCA.

6.2.1.3 Naïve Bayes

Figure 12(a) illustrates the Confusion Matrix of the Naïve Bayes model. Figure 12(b) illustrates the Confusion Matrix of the Naïve Bayes model after PCA is applied on the dataset. Figure 12(c) illustrates the ROC Curve of the Naïve Bayes model while figure 12(d) shows the ROC Curve of the Naïve Bayes model with PCA. In both cases it can be seen from the results that Naïve Bayes has good accuracy but the AUC score is poor. The recall score is less than 0.50 in both cases. Introduction of PCA leads to a decrease in accuracy and precision. However, it leads to an increase in recall, F1 Score and AUC score. This shows that introduction of PCA slightly boosts the performance of the Naïve Bayes model, hence the model is better with PCA.



(a) (b)

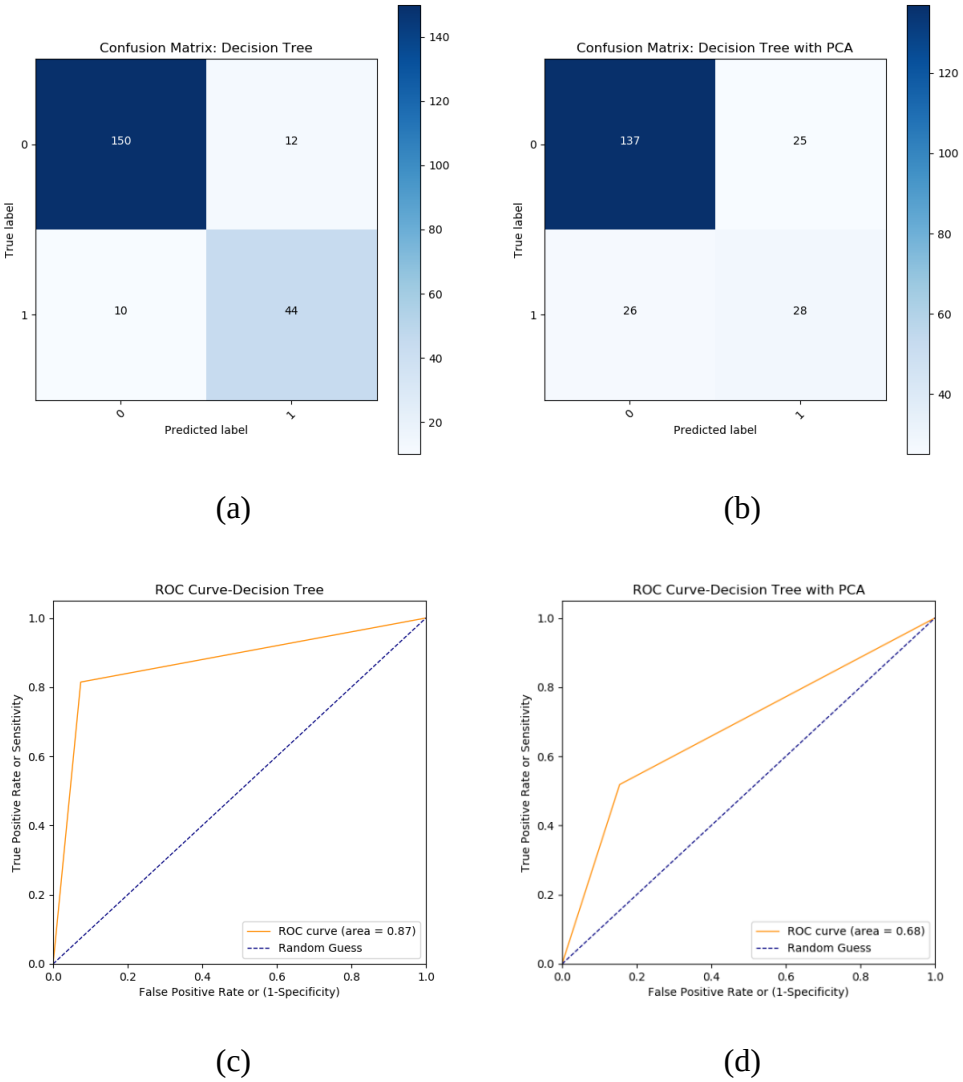


(c) (d)

Fig 12(a). Confusion Matrix of NB, 12(b). Confusion Matrix of NB with PCA, 12(c). ROC Curve of NB and 12(d). ROC Curve of NB with PCA.

6.2.1.4 Decision Tree

Figure 13(a) and Figure 13(b) illustrate the Confusion Matrix of the Decision Tree model with and without PCA applied on the dataset. Figure 13(c) illustrates the ROC Curve of the Decision Tree model while Figure 13(d) shows the ROC Curve of the Decision Tree model with PCA. Decision Tree is an excellent model in the case of Nephropathy since it gives a high accuracy and AUC score. Moreover, the precision, recall and F1 Score are also high with the recall being slightly greater than the precision. Accuracy and AUC decreases slightly. The precision, recall and F1 Score also decrease significantly. This shows that the introduction of PCA, significantly decreases the performance of the Decision Tree model, so the model is better without PCA.



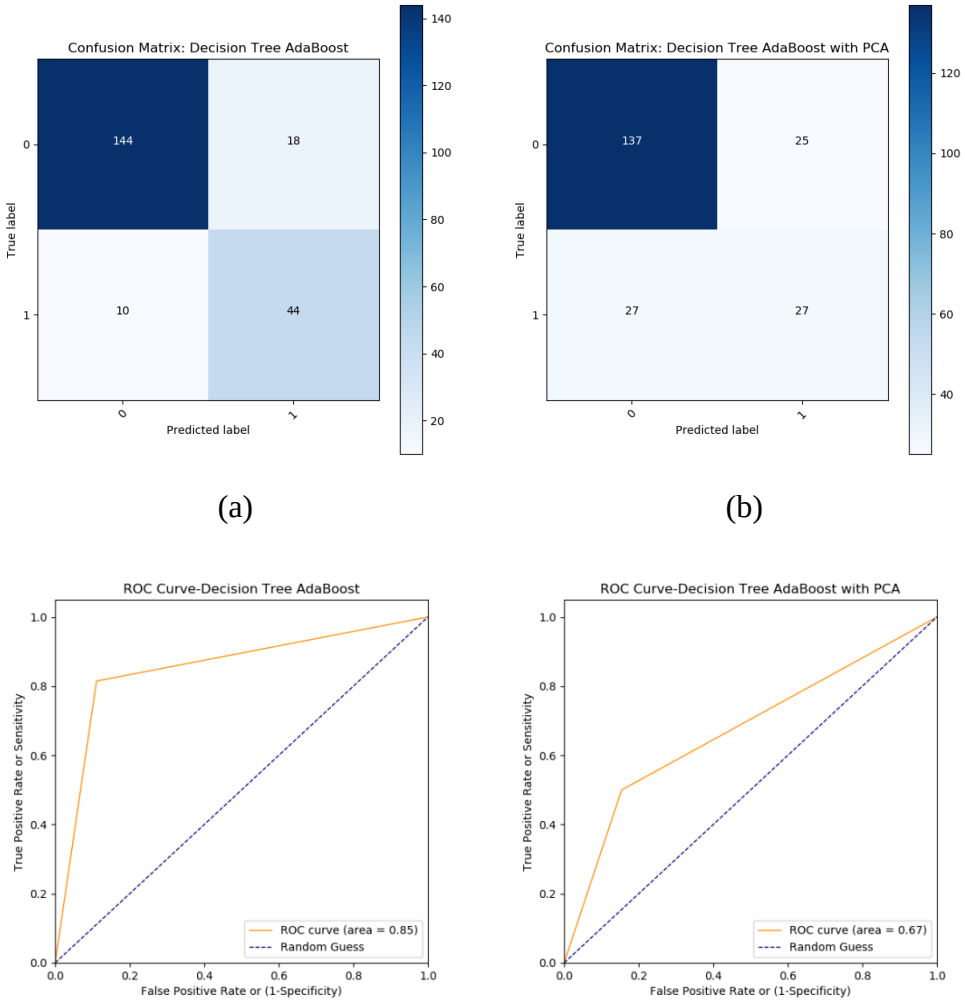
(a) (b)

(c) (d)

Fig 13(a). Confusion Matrix of DT, 13(b). Confusion Matrix of DT with PCA, 13(c). ROC Curve of DT and 13(d). ROC Curve of DT with PCA.

6.2.1.5 Decision Tree with AdaBoost

Figure 14(a) illustrates the Confusion Matrix of the AdaBoosted Decision Tree model. Figure 14(b) illustrates the Confusion Matrix of the AdaBoosted Decision Tree model after PCA is applied on the dataset. Figure 14(c) illustrates the ROC Curve of the AdaBoosted Decision Tree model while figure 14(d) shows the ROC Curve of the AdaBoosted Decision Tree model with PCA.



(a) (b)

(c) (d)

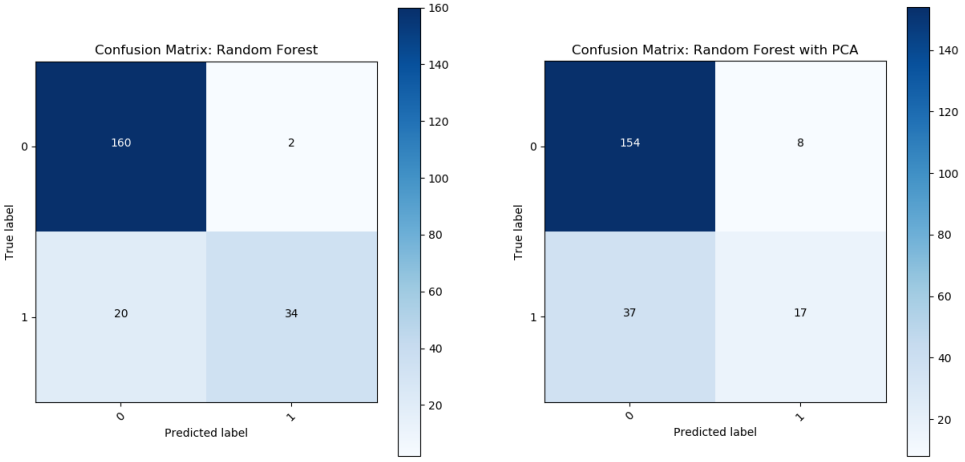
Fig 14(a). Confusion Matrix of DT with AdaBoost, 14(b). Confusion Matrix of AdaBoost DT with PCA, 14(c). ROC Curve of DT with AdaBoost and 13(d). ROC Curve of AdaBoosted DT with PCA.

As mentioned before, Decision Tree is an excellent model in the case of Nephropathy. However, AdaBoost is used on the Decision Tree to see if the performance could be further improved. Instead of boosting the performance though, AdaBoost slightly decreases the Accuracy and AUC score by 0.02. Moreover, the precision and F1 Score also decreases. The recall however, remains the same. When PCA is used before applying AdaBoosted Decision Tree, the performance of the model drops. There is a fall in value of all the performance metrics. Particularly, the AUC score decreases by 0.18. This shows that the introduction of PCA, significantly decreases the performance of the AdaBoosted Decision Tree model, so the model is better without PCA similar to that of the Decision Tree Model.

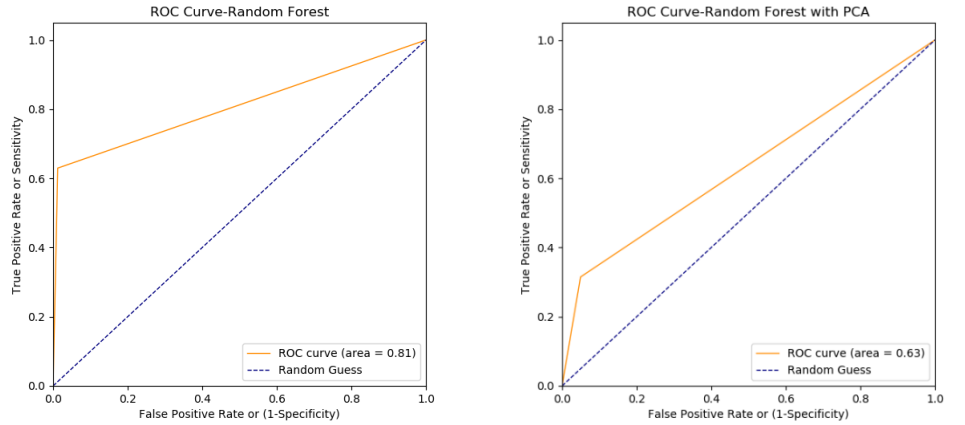
6.2.1.6 Random Forest

Figure 15(a) illustrates the Confusion Matrix of the Random Forest model. Figure 15(b) illustrates the Confusion Matrix of the Random Forest model after PCA is applied on the dataset. Figure 15(c) illustrates the ROC Curve of the Random Forest model while figure 15(d) shows the ROC Curve of the Random Forest model with PCA.

Random Forest is a collection of Decision Trees that aggregate the result of the different trees. This decreases the chance of overfitting, leading to better results. Hence, Random Forest was used to check if it improved the performance of Decision Tree. After comparison, it is seen that accuracy remains same but AUC score decreases quite a lot. Precision increases but recall score drops. Even when PCA is applied beforehand, the results turn out to be even poorer, with accuracy decreasing by 0.10 and AUC score by 0.18. This shows that normal Decision Tree works better than Random Forest. Moreover, Random Forest itself works better without PCA.



(a) (b)



(c) (d)

Fig 15(a). Confusion Matrix of RF, 15(b). Confusion Matrix of RF with PCA, 15(c). ROC Curve of RF and 15(d). ROC Curve of RF with PCA.

6.2.2 Summary of Nephropathy

When PCA is not applied, in terms of accuracy, the best result is produced by Decision Tree and Random Forest without PCA, with a score of 0.89. However, Decision Tree has a recall score of 0.81 which is more than the recall score of Random Forest. Moreover, Decision Tree has an AUC score of 0.87 while Random Forest has an AUC score of 0.81. So even though Random Forest has a significantly greater precision score, Decision Tree is considered to be the best algorithm for the prediction of Nephropathy without PCA.

When PCA is introduced, the accuracy of almost all the algorithms decreases except Logistic Regression. Logistic Regression's accuracy increases from 0.79 to 0.81. The recall and AUC score of Logistic Regression though, decreases. Overall, the AUC score in almost every case decreases below 0.7 when PCA is used. The best result with PCA is given by Logistic Regression with an accuracy of 0.81 and AUC score of 0.73.

Comparing all the possibilities, Decision Tree without PCA is determined to be the combination to predict the onset of Nephropathy.

6.2.3 Cardiovascular Disease

In the model for Cardiovascular Disease, the set of six classification algorithms was used- Logistic Regression (LR), Support Vector Machines (SVM), Naïve Bayes (NB), Decision Tree (DT) with and without AdaBoost and Random Forest (RF). These algorithms were applied on the dataset in four combinations: (i) without PCA, (ii) with PCA, (iii) using Oversampling and (iv) using Oversampling with PCA. In each experiment, the performance was measured using Accuracy, Precision, Recall, Specificity, F1 Score, ROC Curve and AUC Score.

In the following sub-sections, the Confusion Matrix and ROC Curve for every experiment is represented through figures. As shown in Table 1, the Confusion Matrix has four values, which are used to find the Accuracy, Precision, Recall, Specificity and F1 Score to evaluate the performance of each algorithm. The subsequent tables contain all the performance metrics for all the experiments done in the model to predict risk of Cardiovascular Disease.

The table below illustrates the results of different performance metrics for the algorithms to detect risk of Cardiovascular disease without PCA.

Table 4. Cardiovascular Scores without PCA

	Logistic Regression	SVM	Naive Bayes	Decision Tree	Decision Tree AdaBoost	Random Forest
Accuracy	0.91	0.90	0.54	0.88	0.89	0.92
Precision	0.55	0.00	0.18	0.40	0.41	1.00
Recall	0.48	0.00	1.00	0.43	0.39	0.22
F1 Score	0.51	0.00	0.30	0.42	0.40	0.36
Specificity	0.96	0.00	0.49	0.93	0.94	0.00
AUC Score	0.72	0.50	0.74	0.68	0.67	0.61

The table below illustrates the results of different performance metrics for the algorithms to detect risk of Cardiovascular disease with PCA.

Table 5. Cardiovascular Scores with PCA

	Logistic Regression	SVM	Naive Bayes	Decision Tree	Decision Tree AdaBoost	Random Forest
Accuracy	0.91	0.90	0.88	0.87	0.86	0.91
Precision	0.58	0.00	0.38	0.28	0.22	1.00
Recall	0.30	0.00	0.39	0.22	0.17	0.04
F1 Score	0.40	0.00	0.38	0.24	0.20	0.08
Specificity	0.98	0.00	0.93	0.94	0.93	0.00
AUC Score	0.64	0.50	0.66	0.58	0.55	0.52

The table below illustrates the results of different performance metrics for the algorithms to detect risk of Cardiovascular disease using Oversampling.

Table 6. Cardiovascular Scores using Oversampling

	Logistic Regression	SVM	Naive Bayes	Decision Tree	Decision Tree AdaBoost	Random Forest
Accuracy	0.84	0.90	0.63	0.86	0.87	0.90
Precision	0.30	0.43	0.19	0.32	0.37	0.38
Recall	0.49	0.27	0.85	0.43	0.45	0.13
F1 Score	0.38	0.3	0.31	0.37	0.40	0.20
Specificity	0.88	0.96	0.61	0.91	0.92	0.98
AUC Score	0.69	0.61	0.73	0.67	0.69	0.56

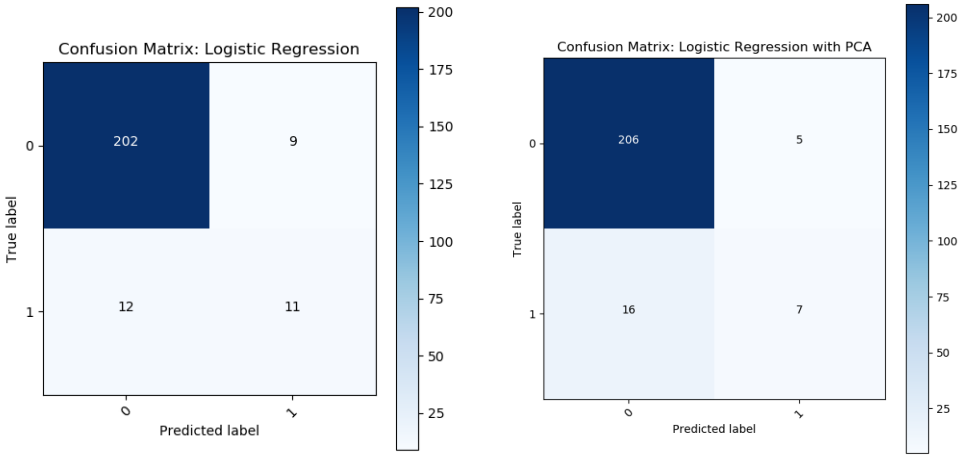
The table below illustrates the results of different performance metrics for the algorithms to detect risk of Cardiovascular disease using Oversampling with PCA.

Table 7. Cardiovascular Scores using Oversampling with PCA

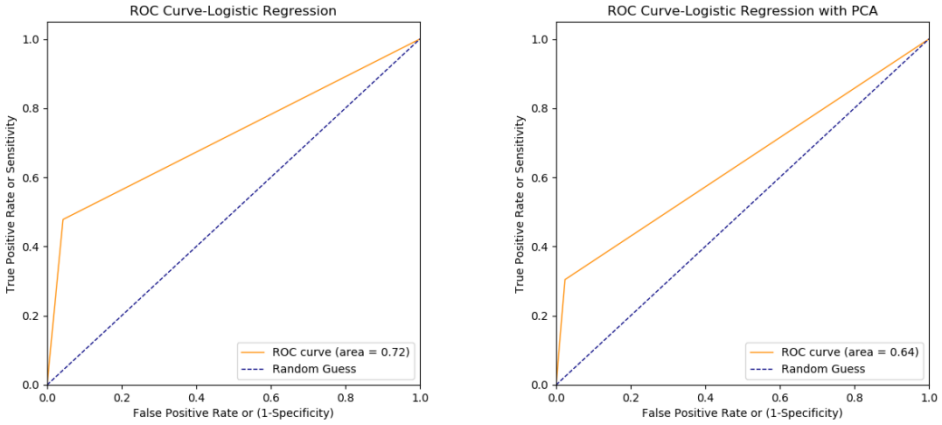
	Logistic Regression	SVM	Naive Bayes	Decision Tree	Decision Tree AdaBoost	Random Forest
Accuracy	0.83	0.91	0.64	0.79	0.79	0.90
Precision	0.30	0.53	0.13	0.19	0.19	0.39
Recall	0.59	0.11	0.47	0.36	0.37	0.17
F1 Score	0.39	0.18	0.20	0.25	0.25	0.24
Specificity	0.85	0.99	0.65	0.84	0.83	0.97
AUC Score	0.72	0.55	0.56	0.60	0.60	0.57

6.2.3.1 Logistic Regression

Figure 16(a) illustrates the Confusion Matrix of the Logistic Regression model. Figure 16(b) illustrates the Confusion Matrix of the Logistic Regression model after PCA is applied on the dataset. Figure 16(c) illustrates the ROC Curve of the Logistic Regression model while Figure 16(d) shows the ROC Curve of the Logistic Regression model with PCA.



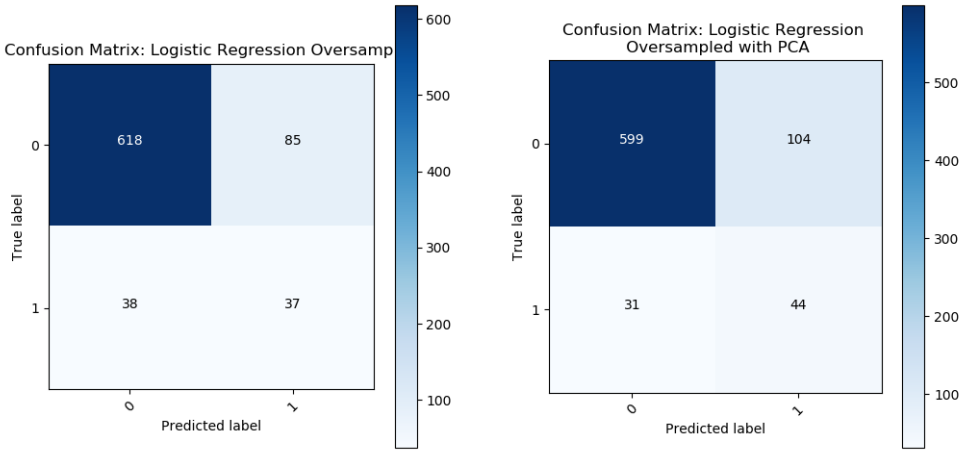
(a) (b)



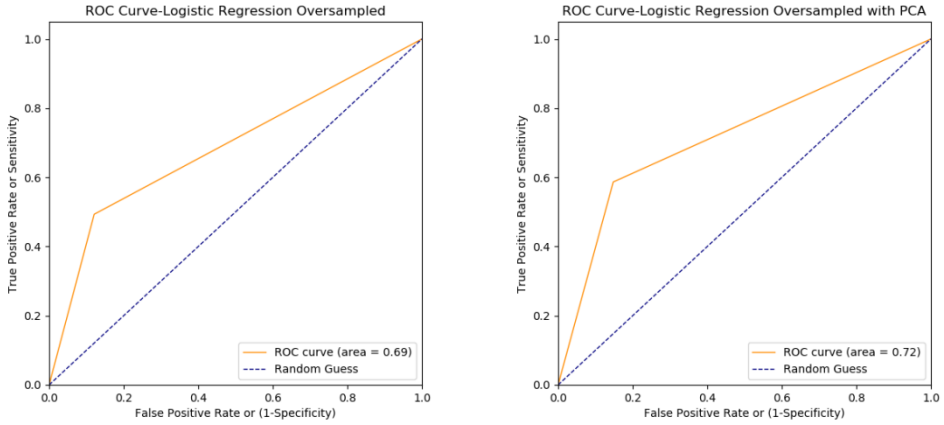
(c) (d)

Fig 16(a). Confusion Matrix of LR, 16(b). Confusion Matrix of LR with PCA, 16(c). ROC Curve of LR and 16(d). ROC Curve of LR with PCA.

Figure 17(a) illustrates the Confusion Matrix of the oversampled Logistic Regression model. Figure 17(b) illustrates the Confusion Matrix of the oversampled Logistic Regression model after PCA is applied on the dataset. Figure 17(c) illustrates the ROC Curve of the Logistic Regression model while Figure 17(d) shows the ROC Curve of the Logistic Regression model with PCA, both on the oversampled dataset.



(a) (b)

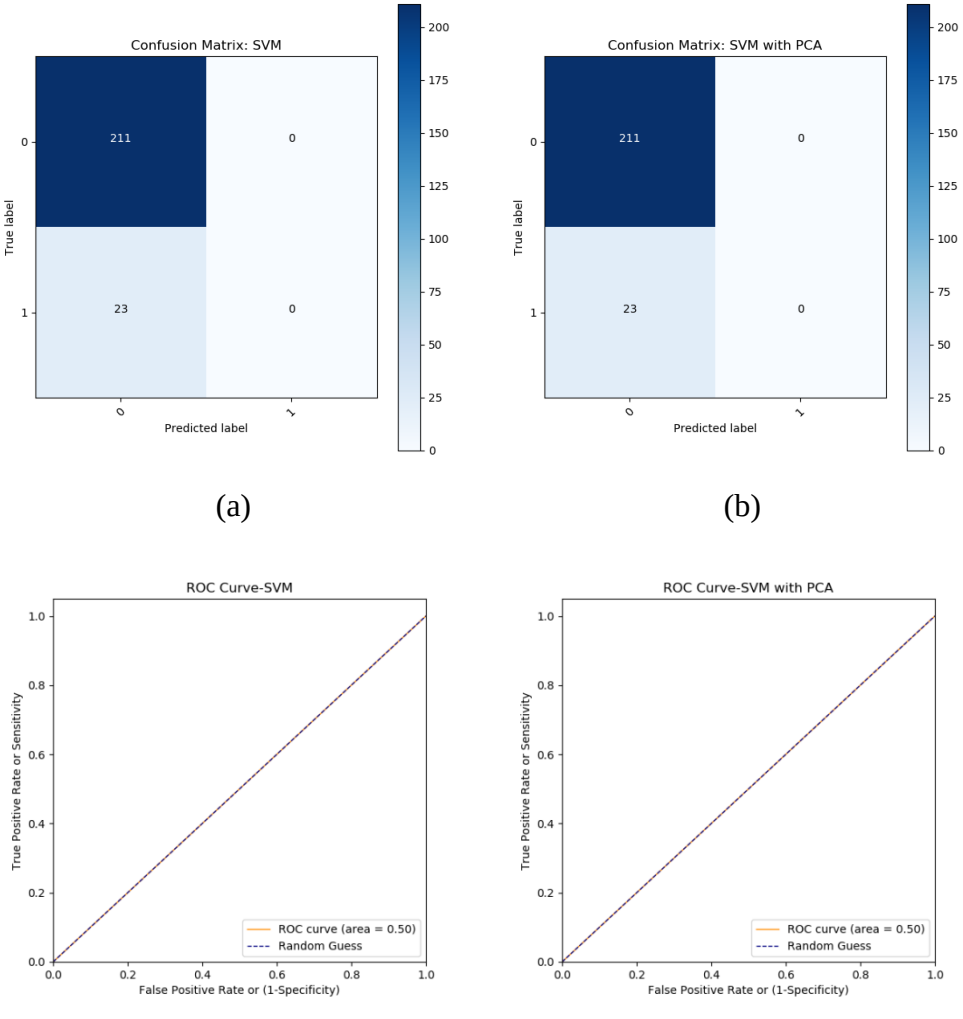


(c) (d)

Fig 17(a). Confusion Matrix of Oversampled LR, 17(b). Confusion Matrix of Oversampled LR with PCA, 17(c). ROC Curve of Oversampled LR and 17(d). ROC Curve of Oversampled LR with PCA.

6.2.3.2 Support Vector Machines (SVM)

Figure 18(a) illustrates the Confusion Matrix of the SVM model. Figure 18(b) illustrates the Confusion Matrix of SVM after PCA is applied on the dataset. Figure 18(c) illustrates the ROC Curve of SVM while Figure 18(d) shows the ROC Curve of SVM with PCA.

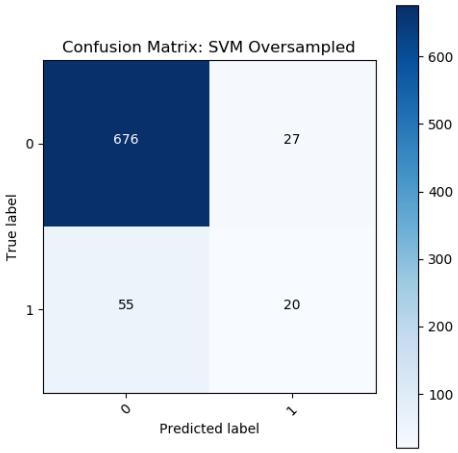


(a) (b)

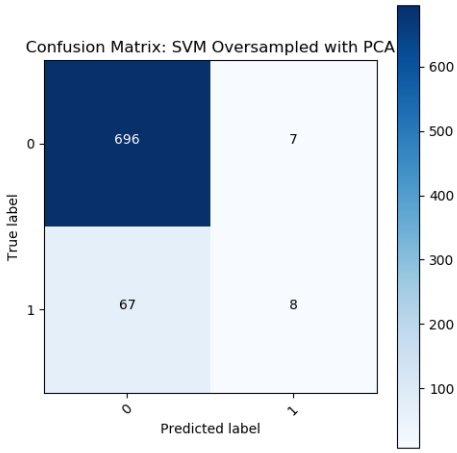
(c) (d)

Fig 18(a). Confusion Matrix of SVM, 18(b). Confusion Matrix of SVM with PCA, 18(c). ROC Curve of SVM and 18(d). ROC Curve of SVM with PCA.

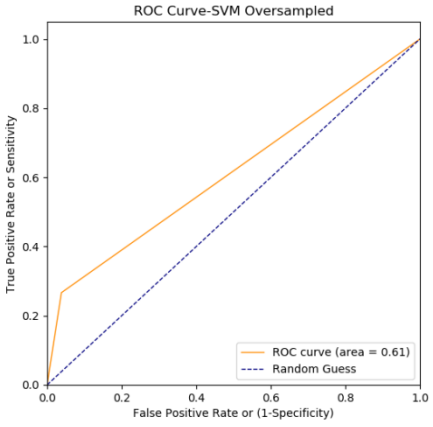
Figure 19(a) illustrates the Confusion Matrix of the oversampled SVM. Figure 19(b) illustrates the Confusion Matrix of the oversampled SVM after PCA is applied on the dataset. Figure 19(c) illustrates the ROC Curve of SVM while Figure 19(d) shows the ROC Curve of SVM with PCA, both on the oversampled dataset.



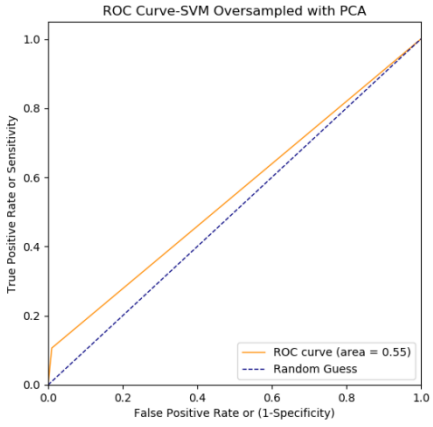
(a)



(b)



(c)

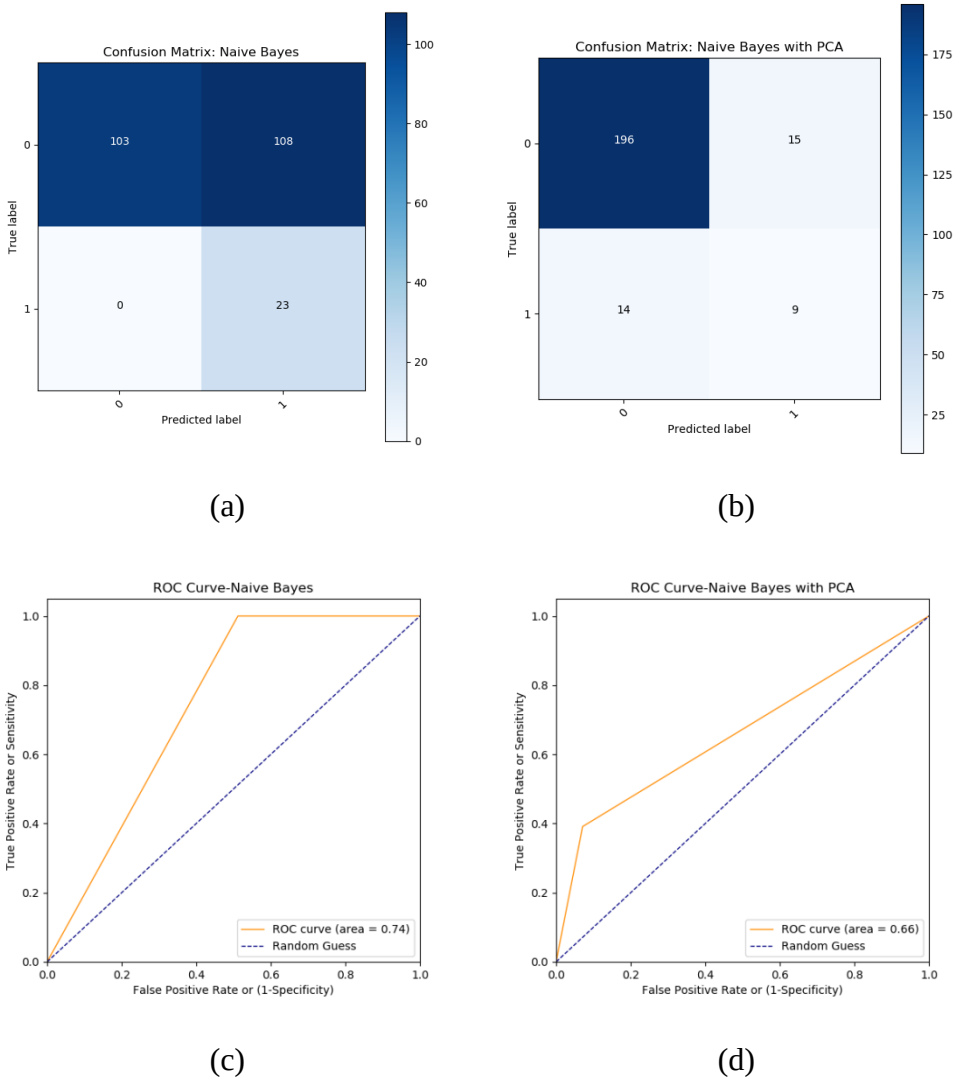


(d)

Fig 19(a). Confusion Matrix of Oversampled SVM, 19(b). Confusion Matrix of Oversampled SVM with PCA, 19(c). ROC Curve of Oversampled SVM and 19(d). ROC Curve of Oversampled SVM with PCA.

6.2.3.3 Naïve Bayes

Figure 20(a) illustrates the Confusion Matrix of the Naïve Bayes model. Figure 20(b) illustrates the Confusion Matrix of Naïve Bayes after PCA is applied on the dataset. Figure 20(c) illustrates the ROC Curve of Naïve Bayes while Figure 20(d) shows the ROC Curve of Naïve Bayes with PCA.

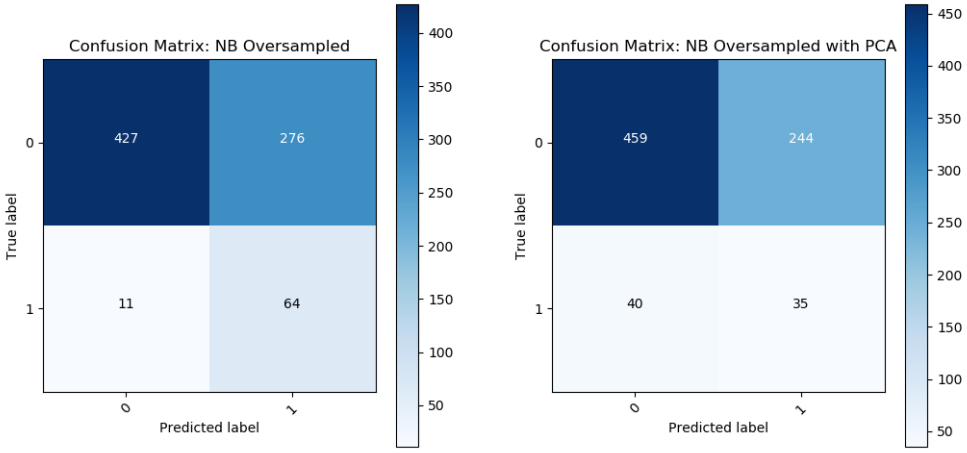


(a) (b)

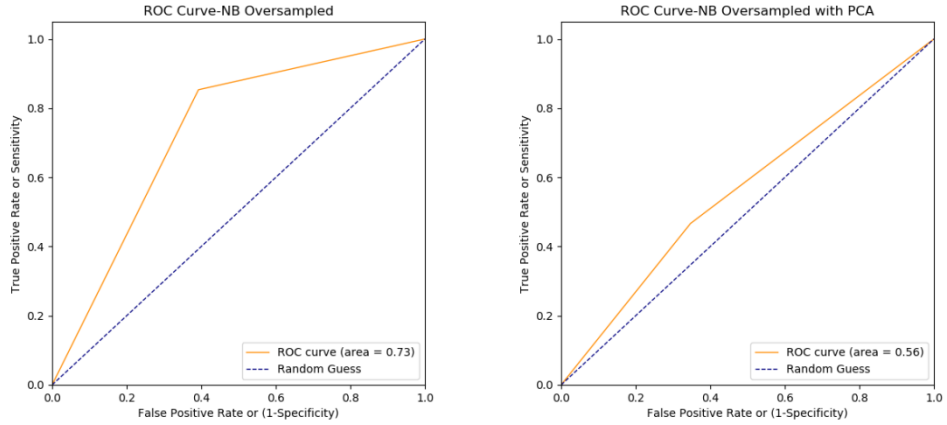
(c) (d)

Fig 20(a). Confusion Matrix of NB, 20(b). Confusion Matrix of NB with PCA, 20(c). ROC Curve of NB and 20(d). ROC Curve of NB with PCA.

Figure 21(a) illustrates the Confusion Matrix of the oversampled Naïve Bayes. Figure 21(b) illustrates the Confusion Matrix of the oversampled Naïve Bayes after PCA is applied on the dataset. Figure 21(c) illustrates the ROC Curve of Naïve Bayes while Figure 21(d) shows the ROC Curve of Naïve Bayes with PCA, both on the oversampled dataset.



(a) (b)

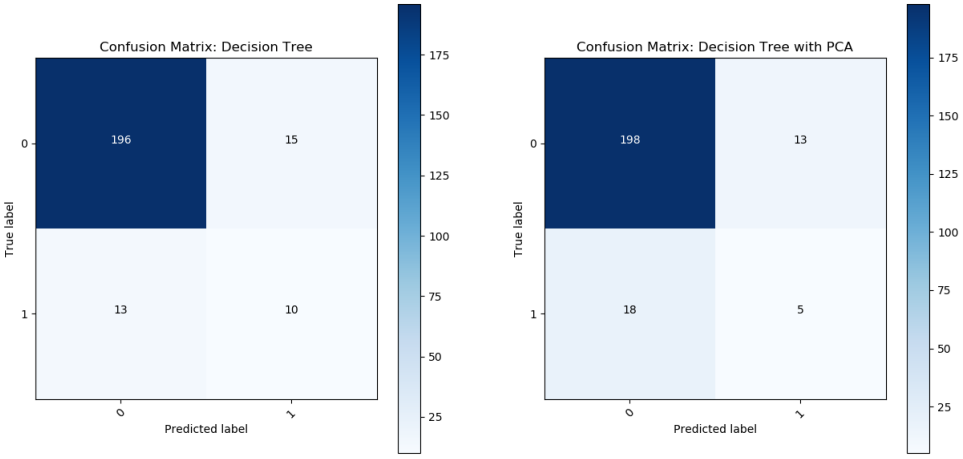


(c) (d)

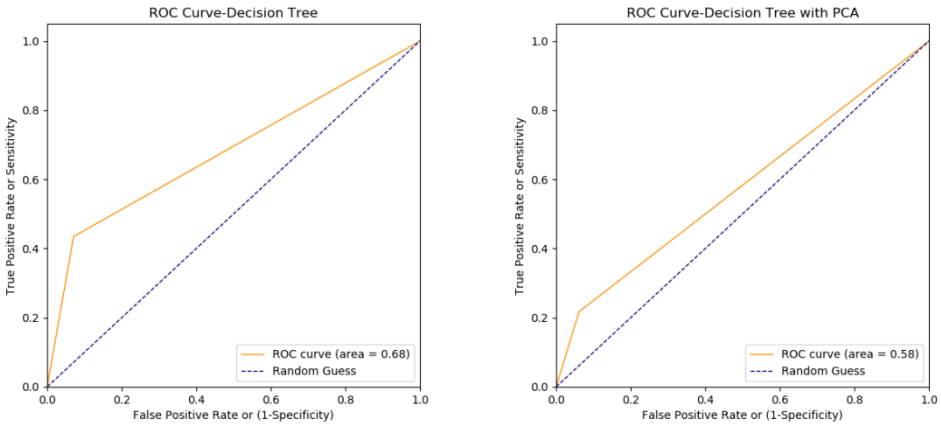
Fig 21(a). Confusion Matrix of Oversampled NB, 21(b). Confusion Matrix of Oversampled NB with PCA, 21(c). ROC Curve of Oversampled NB and 21(d). ROC Curve of Oversampled NB with PCA.

6.2.3.4 Decision Tree

Figure 22(a) illustrates the Confusion Matrix of Decision Tree. Figure 22(b) illustrates the Confusion Matrix of Decision Tree after PCA is applied on the dataset. Figure 22(c) illustrates the ROC Curve of Decision Tree while Figure 22(d) shows the ROC Curve of Decision Tree with PCA.



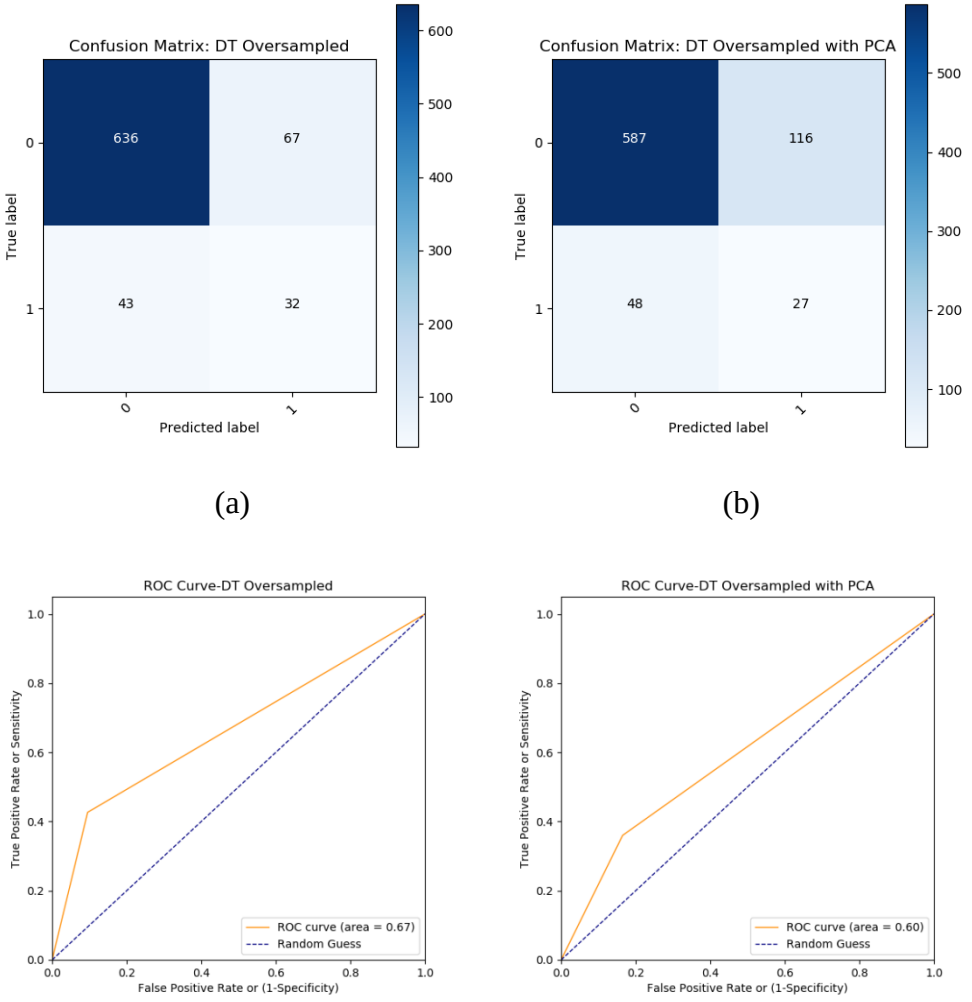
(a) (b)



(c) (d)

Fig 22(a). Confusion Matrix of DT, 22(b). Confusion Matrix of DT with PCA, 22(c). ROC Curve of DT and 22(d). ROC Curve of DT with PCA.

Figure 23(a) illustrates the Confusion Matrix of the oversampled Decision Tree. Figure 23(b) illustrates the Confusion Matrix of the oversampled Decision Tree after PCA is applied on the dataset. Figure 23(c) illustrates the ROC Curve of Decision Tree while Figure 23(d) shows the ROC Curve of Decision Tree with PCA, both on the oversampled dataset.



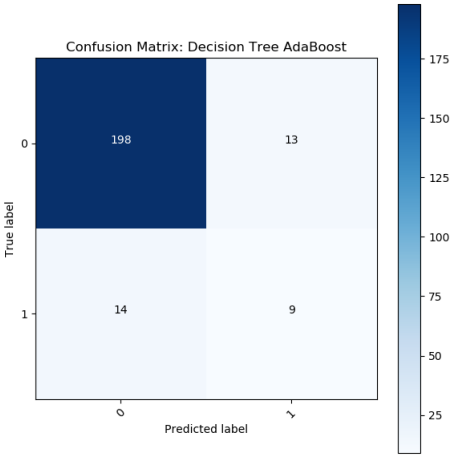
(a) (b)

(c) (d)

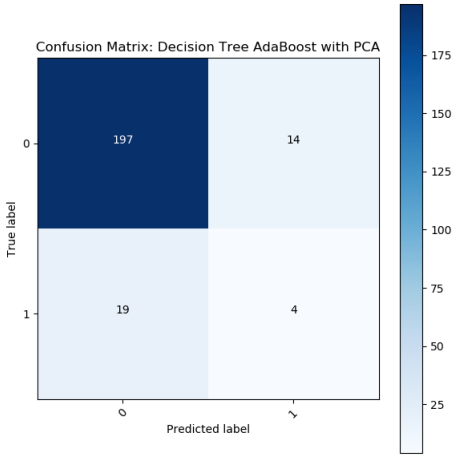
Fig 23(a). Confusion Matrix of Oversampled DT, 23(b). Confusion Matrix of Oversampled DT with PCA, 23(c). ROC Curve of Oversampled DT and 23(d). ROC Curve of Oversampled DT with PCA.

6.2.3.5 Decision Tree with AdaBoost

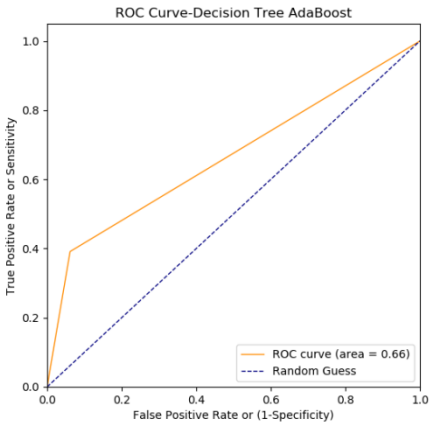
Figure 24(a) illustrates the Confusion Matrix of Decision Tree with AdaBoost. Figure 24(b) illustrates the Confusion Matrix of Decision Tree with AdaBoost after PCA is applied on the dataset. Figure 24(c) illustrates the ROC Curve of Decision Tree with AdaBoost while Figure 24(d) shows the ROC Curve of Decision Tree with AdaBoost with PCA.



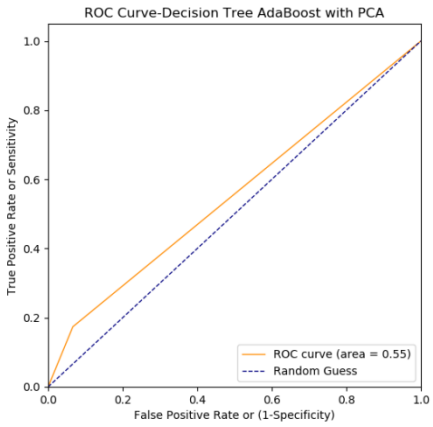
(a)



(b)



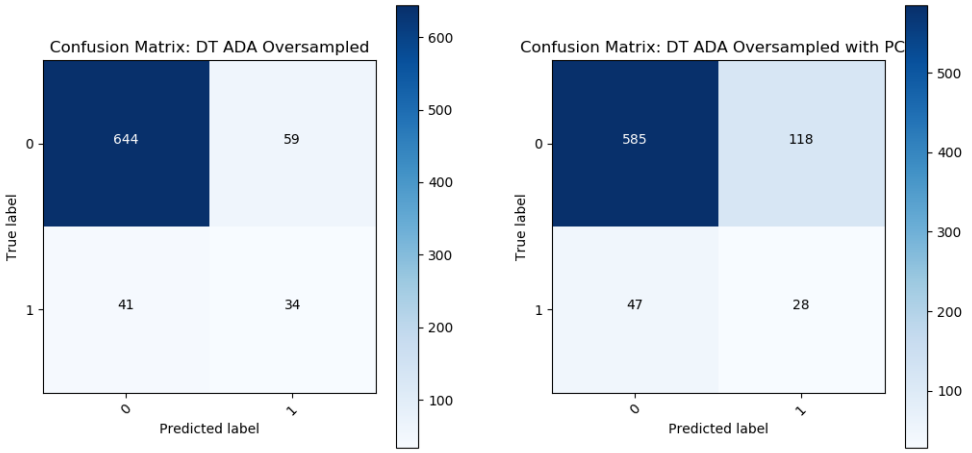
(c)



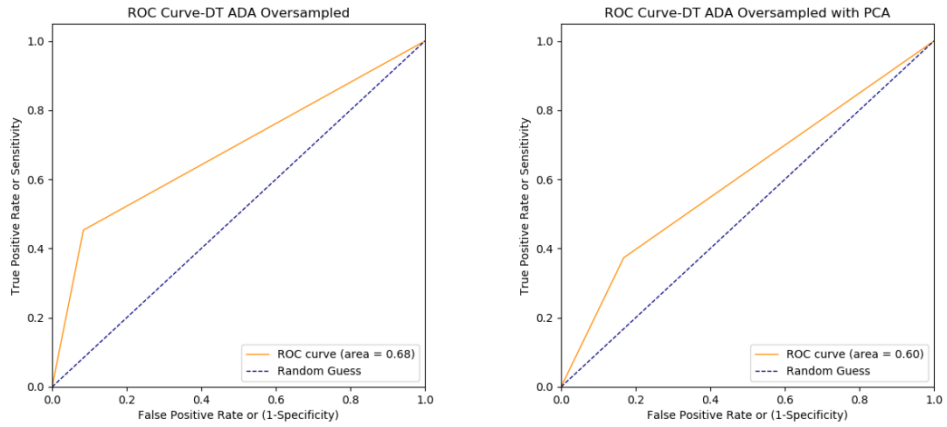
(d)

Fig 24(a). Confusion Matrix of AdaBoosted DT, 24(b). Confusion Matrix of AdaBoosted DT with PCA, 24(c). ROC Curve of AdaBoosted DT and 24(d). ROC Curve of AdaBoosted DT with PCA.

Figure 25(a) illustrates the Confusion Matrix of the oversampled Decision Tree with AdaBoost. Figure 25(b) illustrates the Confusion Matrix of the oversampled Decision Tree with AdaBoost after PCA is applied on the dataset. Figure 25(c) illustrates the ROC Curve of Decision Tree with AdaBoost while Figure 25(d) shows the ROC Curve of AdaBoosted Decision Tree with PCA, both on the oversampled dataset.



(a) (b)

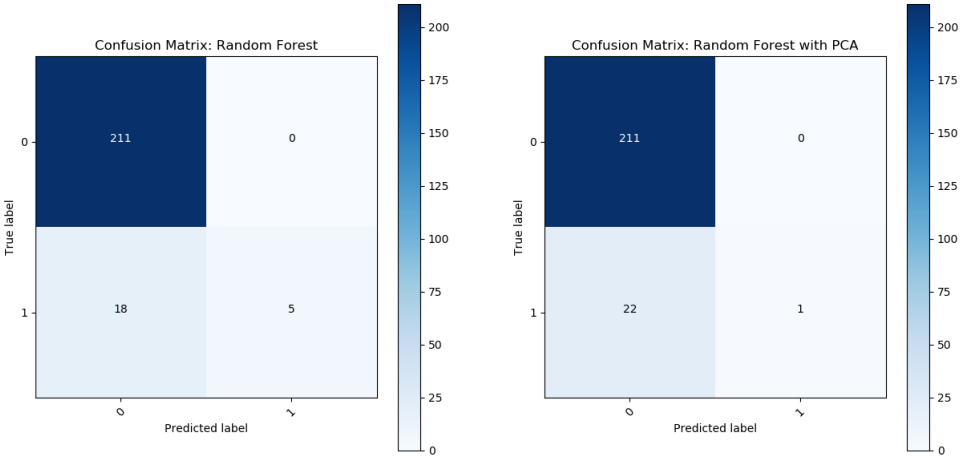


(c) (d)

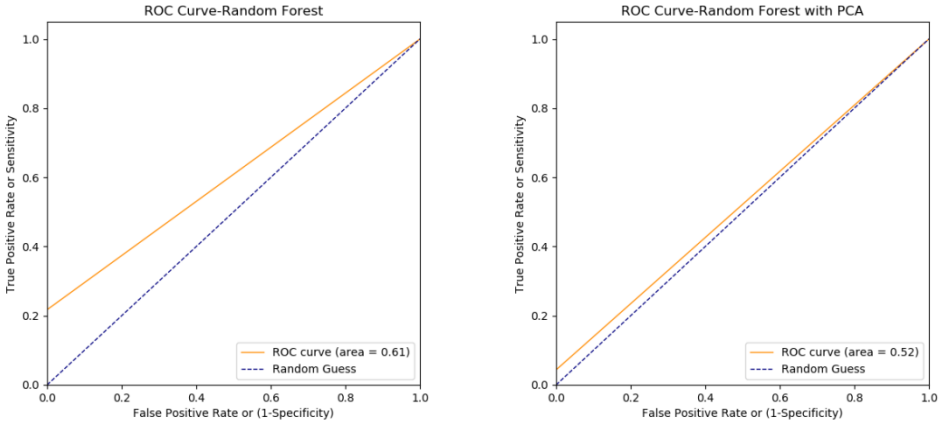
Fig 25(a). Confusion Matrix of AdaBoosted DT with Oversampling, 23(b). Confusion Matrix of AdaBoosted DT with Oversampling plus PCA, 23(c). ROC Curve of AdaBoosted DT with Oversampling and 23(d). ROC Curve of AdaBoosted DT with Oversampling plus PCA.

6.2.3.6 Random Forest

Figure 26(a) illustrates the Confusion Matrix of Random Forest. Figure 26(b) illustrates the Confusion Matrix of Random Forest after PCA is applied on the dataset. Figure 26(c) illustrates the ROC Curve of Random Forest while Figure 26(d) shows the ROC Curve of Random Forest with PCA.



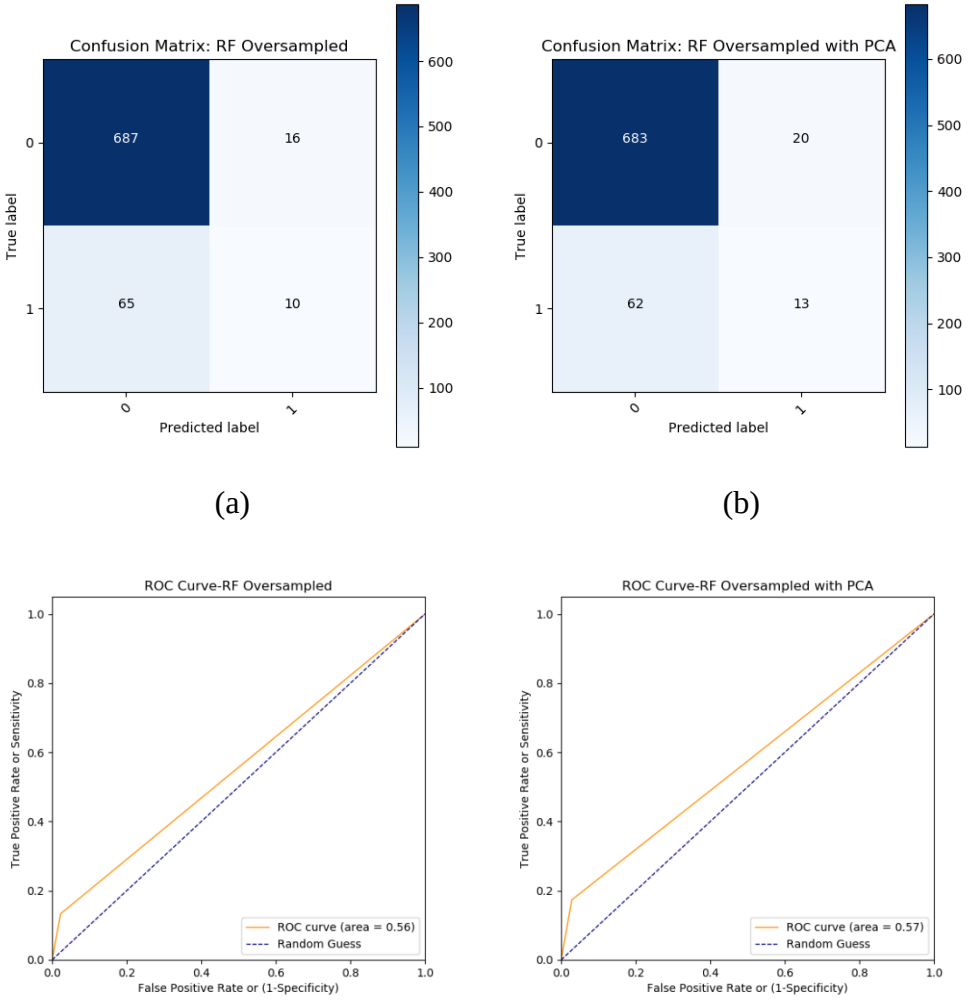
(a) (b)



(c) (d)

Fig 26(a). Confusion Matrix of RF, 26(b). Confusion Matrix of RF with PCA, 26(c). ROC Curve of RF and 26(d). ROC Curve of RF with PCA.

Figure 27(a) illustrates the Confusion Matrix of the oversampled Random Forest. Figure 27(b) illustrates the Confusion Matrix of the oversampled Random Forest after PCA is applied on the dataset. Figure 27(c) illustrates the ROC Curve of Random Forest while Figure 27(d) shows the ROC Curve of Random Forest with PCA, both on the oversampled dataset.



(a) (b)

(c) (d)

Fig 27(a). Confusion Matrix of Oversampled RF, 27(b). Confusion Matrix of Oversampled RF with PCA, 27(c). ROC Curve of Oversampled RF and 27(d). ROC Curve of Oversampled RF with PCA.

6.2.4 Summary of Cardiovascular Disease

In terms of accuracy, the best result is produced by Random Forest without PCA or Oversampling, with a score of 0.92. Logistic Regression is close behind, with a score of 0.91. However, Logistic Regression has a better recall and AUC score than Random Forest. Naïve Bayes though has a better AUC score than both the algorithms mentioned before. It also has a recall score of 1, so even though the accuracy is low, Naïve Bayes is considered to be the best algorithm for the prediction of Cardiovascular Disease without PCA and oversampling.

When PCA is introduced, the accuracy of most of the algorithms remain same except Naïve Bayes. Naïve Bayes' accuracy increases to 0.88 from 0.54 but the recall and AUC score decreases. It still gives the best performance in terms of accuracy and AUC score for Cardiovascular Disease with only PCA.

When only oversampling is used, the best accuracy comes from SVM and Random Forest. Both have poor AUC score of 0.61 and 0.56 respectively. Also their recall is extremely poor. Hence, Naïve Bayes is taken as the best algorithm when oversampling is used since it has an AUC score of 0.73 and recall of 0.85.

When both PCA and oversampling is used, the best performing algorithm is Logistic Regression with an accuracy of 0.83 and AUC score of 0.72.

Comparing all the results, the best combination to predict the risk of Cardiovascular disease is Naïve Bayes without PCA or Oversampling.

6.3 DISCUSSION

To predict the risk of Nephropathy, without PCA, Logistic Regression gives an accuracy of 0.79 and specificity of 0.83. However, the precision, recall and F1 score of this algorithm on the dataset are not satisfactory. Conversely, SVM gives high values for accuracy, precision and specificity, but the recall score falls to just 0.35. On the other hand, Naïve Bayes classifier has an accuracy score of only 0.79 and precision of 0.70, which is comparatively lower than the above mentioned algorithms. Decision Tree was applied in two ways, with and without AdaBoost. While all the performance parameters are above 0.70 and have same recall score, it was observed that Decision Tree works better without AdaBoost. Lastly, Random Forest classifier was implemented on the dataset where the precision and specificity are 0.94 and 0.98 respectively.

While the accuracy is 0.89 which is same as Decision Tree, the recall score is only 0.63 compared to Decision Tree's 0.81. Decision Tree also has the higher AUC score with 0.87 compared to 0.81 of Random Forest. If the performance parameters are considered, Decision Tree has the highest values for accuracy, recall, F1 score and AUC score, while Random Forest has the highest accuracy, precision and specificity.

After implementing PCA on the dataset, the performances of the algorithms change significantly. The overall values for precision, recall, F1 score and AUC score drop for all the algorithms. Logistic Regression gives an accuracy of 0.81 and AUC score of 0.73 which is the highest between all the algorithms when PCA is applied. However, these results are lower than the results found while implementing the algorithms without PCA. Nonetheless, the precision in SVM becomes 1.00 after PCA is applied but the recall score drops to 0.04.

While predicting the risk of Cardiovascular disease, the same set of algorithms was used on the dataset four times; (i) without PCA, (ii) with PCA, (iii) with Oversampling and (iv) with Oversampling after applying PCA.

In the first case, the accuracy of all algorithms except Naïve Bayes is within 0.89 – 0.92. The precision of most of the experiments is poor with SVM's precision being 0. For Random Forest, the precision is 1 but the recall is only 0.22, which is not up to the mark. Similarly, in most instances, the recall is not good, except for Naïve Bayes where it is 1. In this case though, precision is only 0.18. The AUC score is between 0.50 and 0.74 with Naïve Bayes having the highest score of 0.74.

After applying PCA, the results of the classification algorithms do not reach a satisfactory level. While the accuracy is high in all the cases, with Naïve Bayes' accuracy increasing, the precision and recall steps lower compared to the first case. In fact, the recall of Naïve Bayes drops to 0.39 from 1. Moreover, the AUC scores of all algorithms drop, which proves applying just PCA is not a solution to finding the risk of Cardiovascular disease.

The method of Oversampling was implemented on the dataset for the case of Cardiovascular disease after failing to get acceptable results in the first two scenarios. First, only Oversampling was applied on the dataset for the previous set of six algorithms. While there is a noteworthy progress in the results of some algorithms like SVM, Decision Tree and AdaBoosted Decision Tree, the performance of the other algorithms degraded. So when comparing to the previous

results, it is observed that Oversampling does not improve the results enough. Thus, even though the accuracy of some models is high, this approach is also not suitable.

In the fourth scenario, PCA was applied to the dataset initially. Oversampling was then executed on the dataset obtained after PCA was applied. Comparing the results with the previous setup, it is noted that the recall score for Logistic Regression improved. However, it is still poor with the recall score being only 0.59.

It is evident after comparing both with and without PCA, for Nephropathy, the best algorithm is Decision Tree without PCA and AdaBoost. It gives an accuracy of 0.89 and AUC score of 0.87. The recall is also good with 0.81 which is the highest among all the algorithms. The precision and F1 score is also substantial.

On the other hand, for Cardiovascular disease, Naïve Bayes without Oversampling and PCA had the best performance, with an AUC score of 0.74. Even though the accuracy, precision and F1 score were poor, the recall was 1. Which means that, all the patients who had risk of Cardiovascular disease, were diagnosed successfully by the model. Hence, Naïve Bayes without Oversampling and PCA is the best model for prediction of Cardiovascular disease.

PCA was initially applied to remove redundant data and ameliorate the effects of the curse of dimensionality. However, the results showed that, the performance of almost all the algorithms in both cases dropped when PCA was used beforehand. This is was due to data leakage since applying PCA leads to information loss. Therefore it can be inferred that, all the features and their actual values are required for the model to perform well.

After comparing the results of both Nephropathy and Cardiovascular disease, it can be concluded that the system predicts risk of Nephropathy far better than it predicts the risk of Cardiovascular disease. This can be attributed to the fact that most of the feature variables are related to Kidney complications and not Heart complications. Furthermore, the number of positive cases for Cardiovascular disease was very low with only 9.64% of the patients having the disease, compared to 24.9% for Nephropathy.

7 CONCLUSION AND FUTURE WORKS

Diabetes can induce several complications like Nephropathy, Cardiovascular disease, Retinopathy, Neuropathy and many other complex diseases. The paper explains how Machine Learning can be adopted in clinical diagnostics to create systems that use patient-specific information to predict the probability of Diabetes induced complications. This is done using different Machine Learning algorithms under various circumstances. The results show that in case of Nephropathy, the best result is given by Decision Tree without PCA. While for Cardiovascular disease, Naïve Bayes without Oversampling and PCA gave the best performance. The AUC scores were 0.87 and 0.74 respectively. This shows that the system is better at predicting Kidney complications (Nephropathy) than it is at predicting Heart complications (Cardiovascular disease). This happened due to the dataset used had more variables related to Nephropathy than Cardiovascular disease.

Additionally, though the dataset that was used had 164 features, there were only around 780 instances. Thus, due to lack of instances, the predictive models could not work properly in case of prediction of risk of Cardiovascular diseases. Furthermore, there are just 75 patients having Heart disease in the whole dataset, which makes the dataset quite unbalanced and biased. Furthermore, while most features are related to Nephropathy, there were not enough variables present in the dataset so that the model can detect the possibility of Heart disease. Even though the dataset had 164 features, not all features directly impacted the prediction of the diseases. Also, in some cases, the prediction depends on the inter-relationships of the variables, which are not accurately considered in this model in order to minimize complexity to some extent. In actual medical circumstances, all tests are not undertaken, hence values of a lot of variables would be missing. Moreover, the data is of just 1 year which is not quite enough to make a good prediction model as these complications usually develop in the 3^{rd} – 5^{th} year of Diabetes.

As it can be seen that almost all the shortcomings of the system stems from the insufficiency of relevant data, if enough hospitals adopt the system and store their data properly, system like this one can be used more efficiently and effectively. In future, a predictive model like this one can be used for prognosis, diagnosis and treatment planning of patients within clinical information systems. The model can assist doctors in making automatic preliminary diagnosis of the

complications so they can invest more time on the patients who have far greater chances of developing complications. Furthermore, tools can be developed that will also help the patients get an idea about their current state without needing to visit the doctor every time.

The motivation behind this project was to help the diabetic patients lead a better life. Additionally, while there are a lot of research works done on this topic, there has been only a limited number of real-life systems. Hence, with proper data and Machine Learning techniques, it is possible to build an out-and-out application in the near future rather than restricting the idea into research only.

With prevention or cure of Diabetes yet to be found, it is an alarming issue for people across the world. It is estimated to rise to up to 629 million diabetic patients by 2045. Keeping the thoughts of such a significant portion of the world population, it is vital to come up with systems that can aid in providing a better lifestyle to these people. It is important to diagnose and monitor the complications induced by Diabetes in the early stages since at times they are much more harmful than the disease itself. If these complications can be predicted early, their effects can be significantly reduced. This paper explores the field of Machine Learning as a means for the prediction of Kidney and Heart complications which can ultimately result in a complete clinical system helping both patients and doctors in the future.

REFERENCES

[1] Heisler, M., Piette, J. D., Spencer, M., Kieffer, E., & Vijan, S. (2005). The relationship between knowledge of recent HbA1c values and Diabetes care understanding and self-management. Diabetes care, 28(4), 816-822.

[2] World Health Organization. (1999). Definition, diagnosis and classification of Diabetes mellitus and its complications: report of a WHO consultation. Part 1, Diagnosis and classification of Diabetes mellitus (No. WHO/NCD/NCS/99.2). Geneva: World health organization.

[3] American Diabetes Association. (2014). Diagnosis and classification of Diabetes mellitus. Diabetes care, 37(Supplement 1), S81-S90.

[4] Gregg, E. W., Li, Y., Wang, J., Rios Burrows, N., Ali, M. K., Rolka, D., et al. (2014). Changes in Diabetes-related complications in the United States, 1990–2010. New England Journal of Medicine, 370(16), 1514-1523.

[5] Deshpande, A. D., Harris-Hayes, M., & Schootman, M. (2008). Epidemiology of Diabetes and Diabetes-related complications. Physical therapy, 88(11), 1254-1264.

[6] Yun, 15 July 2018, Chinese AI Beats Doctors in Diagnosing Brain Tumors.

[7] Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V., & Fotiadis, D. I. (2015). Machine Learning applications in cancer prognosis and prediction. Computational and structural biotechnology journal, 13, 8-17.

[8] Koutsouleris, N., Meisenzahl, E. M., Davatzikos, C., Bottlender, R., Frodl, T., Scheuerecker, J., et al. (2009). Use of neuroanatomical pattern classification to identify subjects in at-risk mental states of psychosis and predict disease transition. Archives of general psychiatry, 66(7), 700-712.

[9] Bradley, A. P. (1997). The use of the area under the ROC Curve in the evaluation of Machine Learning algorithms. Pattern recognition, 30(7), 1145-1159.

[10] Barakat, N., Bradley, A. P., & Barakat, M. N. H. (2010). Intelligible Support Vector Machines for diagnosis of Diabetes mellitus. IEEE transactions on information technology in biomedicine, 14(4), 1114-1120.

[11] Dagliati, A., Marini, S., Sacchi, L., Cogni, G., Teliti, M., Tibollo, V., et al. (2018). Machine Learning methods to predict Diabetes complications. Journal of Diabetes science and technology, 12(2), 295-302.

[12] Cho, B. H., Yu, H., Kim, K. W., Kim, T. H., Kim, I. Y., & Kim, S. I. (2008). Application of irregular and unbalanced data to predict diabetic Nephropathy using visualization and feature selection methods. Artificial intelligence in medicine, 42(1), 37-53.

[13] Tomonaga, O. (2017, April 27). JAMP_DATA0722figshaer.xlsx (Version 1). [Retrieved from: doi.org/10.6084/m9.figshare.4924037.v1].

[14] Cercato, C., Mancini, M. C., Arguello, A. M. C., Passos, V. Q., Villares, S. M. F., & Halpern, A. (2004). Systemic hypertension, Diabetes mellitus, and dyslipidemia in relation to body mass index: evaluation of a Brazilian population. Revista do Hospital das Clínicas, 59(3), 113-118.

[15] Han, S. H., Nicholls, S. J., Sakuma, I., Zhao, D., Koh, K. K. (2016). Hypertriglyceridemia and Cardiovascular Diseases: Revisited. Korean Circ J, 46(2), 135–144. Doi: 10.4070/kcj.2016.46.2.135.

[16] Gross, J. L., De Azevedo, M. J., Silveiro, S. P., Canani, L. H., Caramori, M. L., & Zelmanovitz, T. (2005). Diabetic Nephropathy: diagnosis, prevention, and treatment. Diabetes care, 28(1), 164-176.

[17] Mogensen, C. E. (1987). Microalbuminuria as a predictor of clinical diabetic Nephropathy. Kidney international, 31(2), 673-689.

[18] Jensen, J. S., Clausen, P., Borch-Johnsen, K., Jensen, G., & Feldt-Rasmussen, B. (1997). Detecting microalbuminuria by urinary albumin/creatinine. Nephrol Dial Transplant, 12(2), 6-9.

[19] Dabla, P. K. (2010). Renal function in diabetic Nephropathy. World journal of Diabetes, 1(2), 48.

[20] Sterne, J. A., White, I. R., Carlin, J. B., Spratt, M., Royston, P., Kenward, M. G., et al. (2009). Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls. Bmj, 338, b2393.

[21] Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. R news, 2(3), 18-22.

[22] Oshiro, T. M., Perez, P. S., & Baranauskas, J. A. (2012, July). How many trees in a Random Forest?. In International Workshop on Machine Learning and Data Mining in Pattern Recognition (pp. 154-168). Springer, Berlin, Heidelberg.

[23] Stekhoven, D. J., & Bühlmann, P. (2011). MissForest—non-parametric missing value imputation for mixed-type data. Bioinformatics, 28(1), 112-118.

[24] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning research, 12(Oct), 2825-2830.

[25] Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: a review and recent developments. Phil. Trans. R. Soc. A, 374(2065), 20150202.

[26] Perveen, S., Shahbaz, M., Keshavjee, K., & Guergachi, A. (2018). A Systematic Machine Learning Based Approach for the Diagnosis of Non-Alcoholic Fatty Liver Disease Risk and Progression. Scientific reports, 8(1), 2112.

[27] Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. Intelligent data analysis, 6(5), 429-449.

[28] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357.

[29] Hawkins, D. M. (2004). The problem of overfitting. Journal of chemical information and computer sciences, 44(1), 1-12.

[30] Gholami, V., Chau, K. W., Fadaee, F., Torkaman, J., & Ghaffari, A. (2015). Modeling of groundwater level fluctuations using dendrochronology in alluvial aquifers. Journal of hydrology, 529, 1060-1069.

[31] Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2011). Classifier chains for multi-label classification. Machine Learning, 85(3), 333.

[32] Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied Logistic Regression* (Vol. 398). John Wiley & Sons.

[33] Collins, M., Schapire, R. E., & Singer, Y. (2002). Logistic Regression, AdaBoost and Bregman distances. *Machine Learning*, *48*(1-3), 253-285.

[34] Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support Vector Machines. *IEEE Intelligent Systems and their applications*, *13*(4), 18-28.

[35] Steinwart, I., & Christmann, A. (2008). *Support Vector Machines*. Springer Science & Business Media.

[36] Zhang, H. (2004). The optimality of Naive Bayes. AA, 1(2), 3.

[37] Loh, W. Y. (2011). Classification and regression trees. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1(1), 14-23.

[38] Breiman, L. (2017). Classification and regression trees. Routledge.

[39] Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, 55(1), 119-139.

[40] Hastie, T., Rosset, S., Zhu, J., & Zou, H. (2009). Multi-class AdaBoost. Statistics and its Interface, 2(3), 349-360.