**Diabetes Prediction Using Ensembling of Different Machine Learning Classifiers**

A PROJECT REPORT

Submitted in partial fulfilment of requirements to

CS452 PROJECT-II

IV/IV B.Tech CSE(VII Semester)

Submitted By

**Batch 1**

#### **B.Sushma Sri(Y18CS016)**

#### **D.Sri Harsha Chowdary(Y18CS034)**

#### **A.Chanikya Subhash(Y18CS003)**

Under the guidance of

#### **Dr.M.Sreelatha**

Prof & HOD,CSE



**November 2021**

#### **Department of Computer Science and Engineering**

**R.V.R. & J.C. COLLEGE OF ENGINEERING**

(Autonomous)

Approved by AICTE :: Affiliated to Acharya Nagarjuna University Chowdavaram, Guntur - 522019, Andhra Pradesh, India

**Department of Computer Science and Engineering**



**CERTIFICATE**

This is to certify that project work titled **“Diabetes Prediction Using Ensembing of Different Machine Learning Classifiers “**is the study presented and submitted by **B.Sushma Sri(Y18CS016), D.Sri Harsha Chowdary(Y18CS034), A.Chanikya Subhash (Y18CS003),**

in partial fulfillment of the requirements for the **CS452 Project II** of

Bachelor of Technology during 2020-2021.

**Dr. M.Sreelatha Smt.K.Venkata Ramana**

Term Paper Guide Term Paper Incharge

Prof.& HOD,CSE Associate Professor ,CSE

**Dr.M.Sreelatha**

Prof. and HOD,CSE

## **ACKNOWLEDGEMENT**

The successful completion of any task would be incomplete without a proper suggestions, guidance and environment. Combination of these three factors acts like backbone to our Project “**Diabetes Prediction Using Ensembling of Machine Learning Classifiers.**”

We are very glad to express our special thanks to **Dr. M.Sreelatha** ,Prof. and HOD,CSE and Term Paper Guide, who helped us to understand the topic and also for her valuable advices in preparing this term paper report.

We would like to express our special thanks of gratitude to **Smt. K.Venkata Ramana**, Term Paper Incharge, for her support.

We are very much thankful to **Dr. K.Ravindra**, Principal of RVR & JC COLLEGE OF ENGINEERING, Guntur for providing the supportive environment.

Finally we submit our reserves thanks to lab staff in Department of Computer Science and Engineering and to all our friends for their cooperation during this project work preparation.

**B. Sushma Sri (Y18CS016)**

**D. Sri Harsha Chowdary (Y18CS034)**

**A. Chanikya Subhash (Y18CS003)**

**ABSTRACT**

Diabetes, also known as chronic illness, is a group of metabolic diseases due to a high level of sugar in the blood over a long period .The risk factor and severity of diabetes can be reduced significantly if the precise early prediction is possible .The robust and accurate prediction of diabetes is highly challenging due to the limited number of labeled data and also the presence of outliers or missing values in the diabetes datasets.

A robust framework for diabetes prediction where the outlier rejection, filling the missing values, data standardization, feature selection, K-fold cross-validation, and different Machine Learning (ML) classifiers(k-nearest Neighbor, Decision Trees, Random Forest, AdaBoost, Naive Bayes, and XGBoost) and Multilayer Perceptron (MLP) were employed.

The weighted ensembling of different ML models is also proposed, to improve the prediction of diabetes where the weights are estimated from the corresponding Area Under ROC Curve (AUC) of the ML model.

AUC is chosen as the performance metric, which is then maximized during hyperparameter tuning using the grid search technique. All the experiments, in this literature, were conducted under the same experimental conditions using the Pima Indian Diabetes Dataset.

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **CHAPTER** | **DESCRIPTION** | **PAGE NO** |
|  | **Abstract** | **iv** |
|  | **List of Figures** | **vi** |
|  | **List of Tables** | **vii** |
|  | **List of Abbrevations** | **viii** |
| **1** | **INTRODUCTION** | **1** |
|  | 1.1 Background | 1 |
|  | 1.2 Problem Statement | 5 |
|  | 1.3 Objectives | 5 |
|  | 1.4 Need of Present Study | 6 |
| **2** | **LITERATURE REVIEW** | **8** |
|  | 2.1 Global Estimates of Diabetes | 8 |
|  | 2.2 Machine Learning Models | 11 |
|  | 2.3 Neural Network Models | 17 |
|  | 2.4 Genetic Programming | 18 |
|  | 2.5 Single Machine Learning Algorithms | 19 |
|  | 2.6 Comparing Classifiers | 22 |
|  | 2.7 Models with Preprocessing | 24 |
|  | 2.8 Ensembling Models | 26 |
| **3** | **METHODOLOGIES** | **30** |
|  | 3.1 Architecture | 30 |
|  | 3.2 Modules | 32 |
|  | 3.3 Dataset | 58 |
| **4** | **DISCUSSION ON PROPOSED OUTPUT** | **62** |
| **5** | **CONCLUSION AND FUTURE WORK** | **65** |
| **6** | **REFERENCES** | **67** |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **DESCRIPTION** | **PAGE NO** |
| 3.1.1. | Block diagram of Diabetes Prediction | 30 |
| 3.2.1. | Attributes in the PIMA Indian Diabetes Dataset | 32 |
| 3.2.2.1. | Partitioning of the PID dataset for KCV | 40 |
| 3.2.3.1.1. | Working of k-Nearest Neighbors | 42 |
| 3.2.3.2.1. | Construction of Decision Tree | 44 |
| 3.2.3.3.1. | Stump of AdaBoost | 46 |
| 3.2.3.4.1. | Building Random Forest | 49 |
| 3.2.3.6.1. | XGBoost Features | 53 |
| 3.2.5.1. | The MLP architecture | 58 |
| 3.3.1. | PIMA Indian Diabetes Dataset | 61 |
| 4.1.1. | Confusion Matrix | 62 |
| 4.1.2. | Area Under Curve | 63 |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **DESCRIPTION** | **PAGE NO** |
| 3.2.3.6.1. | Different ML Models with hyper parameters | 55 |
| 3.3.1. | The overview of Diabetic Patient Cohort | 59 |

**List of Abbreviations**

**S.No Abbrevation Full Form**

1. PID PIMA Indians Diabetes Dataset

2. PCA Principle Component Analysis

3. ICA Independent Component Analysis

4. KCV K-Fold Cross-Validation

5. k-NN k-Nearest Neighbour

6. DT Decision Tree

7. AB AdaBoost (Adaptive Boosting)

8. RF Random Forest

9. NB Naïve Bayes

10. XB Extreme Gradient Boosting

11. MLP Multi Layer Perceptron

12. ROC Receiver Operating Characteristics

13. AUC Area Under Curve

14. FOR False Omission Rate

15. DOR Diagnostic Odds Ratio

16. ML Machine Learning

1. **INTRODUCTION**
   1. **Background**

Diabetes, also known as chronic illness, is a group of metabolic diseases due to a high level of sugar in the blood over a long period. The risk factor and severity of diabetes can be reduced significantly if the precise early prediction is possible. The robust and accurate prediction of diabetes is highly challenging due to the limited number of labeled data and also the presence of outliers (or missing values) in the diabetes datasets.

**1.1.1** **Diabetes:** Diabetes is a very familiar word in the present world and crucial challenges in both developed and developing countries. The insulin hormone in the body produced by the pancreas allows glucose to pass from the food into the bloodstream. The lack of that hormone due to malfunctioning of the pancreas forms diabetes which can result in coma, renal and retinal failure, pathological destruction of pancreatic beta cells, cardiovascular dysfunction, cerebral vascular dysfunction, peripheral vascular diseases, sexual dysfunction, joint failure, weight loss, ulcer, and pathogenic effects on immunity.

Research on diabetes patients demonstrates that diabetes among adults (over 18 years old) has risen from 4.7 % to 8.5 % in 1980 to 2014 respectively and rapidly growing up in second and third world countries. Statistical results in 2017 show that 451 million people were living with diabetes worldwide, which will increase to 693 million by 2045. Another statistical study shows the severity of diabetes, where they reported that half a billion people have diabetes worldwide, and the number will increase to 25 % and 51 % respectively in 2030 and 2045.

Without ongoing, careful management, diabetes can lead to a build up of sugars in the blood, which can increase the risk of dangerous complications, including [stroke](https://www.medicalnewstoday.com/articles/7624.php) and [heart disease](https://www.medicalnewstoday.com/articles/237191.php). Different kinds of diabetes can occur, and managing the condition depends on the type. Not all forms of diabetes stem from a person being overweight or leading an inactive lifestyle. In fact, some are present from childhood.

Three major diabetes types can develop: Type 1, type 2, and gestational diabetes.

**Type I diabetes:** Also known as juvenile diabetes, this type occurs when the body fails to produce [insulin](https://www.medicalnewstoday.com/info/diabetes/whatisinsulin.php). People with [type I diabetes](https://www.medicalnewstoday.com/info/diabetes/type1diabetes.php) are insulin-dependent, which means they must take artificial insulin daily to stay alive.

**Type 2 diabetes:** [Type 2 diabetes](https://www.medicalnewstoday.com/info/diabetes/type2diabetes.php) affects the way the body uses insulin. While the body still makes insulin, unlike in type I, the cells in the body do not respond to it as effectively as they once did. This is the most common type of diabetes, according to the National Institute of Diabetes and Digestive and Kidney Diseases, and it has [strong linksTrusted Source](https://www.niddk.nih.gov/health-information/diabetes/overview/what-is-diabetes) with [obesity](https://www.medicalnewstoday.com/info/obesity/how-much-should-i-weigh.php).

**Gestational diabetes:** This type occurs in women during pregnancy when the body can become less sensitive to insulin. Gestational diabetes does not occur in all women and usually resolves after giving birth.

Insulin allows the glucose from a person’s food to access the cells in their body to supply energy. Insulin resistance is usually a result of the following cycle. A person has genes or an environment that make it more likely that they are unable to make enough insulin to cover how much glucose they eat. The body tries to make extra insulin to process the excess blood glucose. The pancreas cannot keep up with the increased demands, and the excess blood sugar starts to circulate in the blood, causing damage. Over time, insulin becomes less effective at introducing glucose to cells, and blood sugar levels continue to rise.

Steps a person can take to embrace a lifestyle with diabetes include: Eating a diet high in fresh, nutritious foods, including whole grains, fruits, vegetables, lean proteins, low-fat dairy, and healthy fat sources, such as nuts. Avoiding high-sugar foods that provide empty [calories](https://www.medicalnewstoday.com/articles/245588.php), or calories that do not have other nutritional benefits, such as sweetened sodas, fried foods, and high-sugar desserts. Refraining from drinking excessive amounts of alcohol or keeping intake to less than one drink a day for women or two drinks a day for men. Engaging in at least 30 minutes exercise a day on at least 5 days of the week, such as of walking, aerobics, riding a bike, or swimming. Recognizing signs of low blood sugar when exercising, including dizziness, confusion, weakness, and profuse sweating.

**1.1.2 Machine Learning:** Machine learning is a type of artificial intelligence that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning [algorithms](https://whatis.techtarget.com/definition/algorithm) use historical data as input to predict new output values.

Machine learning is important because it gives enterprises a view of trends in customer behaviour and business operational patterns, as well as supports the development of new products. Many of today's leading companies, such as Facebook, Google and Uber, make machine learning a central part of their operations. Machine learning has become a significant competitive differentiator for many companies.

Classical machine learning is often categorized by how an algorithm learns to become more accurate in its predictions. There are four basic approaches: [supervised](https://searchenterpriseai.techtarget.com/definition/supervised-learning) learning, [unsupervised](https://whatis.techtarget.com/definition/unsupervised-learning) learning, semi-supervised learning and reinforcement learning. The type of algorithm data scientists choose to use depends on what type of data they want to predict.

**1.1.3 Ensembling:** Ensembling is a technique of combining two or more algorithms of similar or dissimilar types called base learners. This is done to make a more robust system which incorporates the predictions from all the base learners.

Ensembling is a proven method for improving the accuracy of the model and works in most of the cases. It is the key ingredient for winning almost all of the machine learning hackathons. Ensembling makes the model more robust and stable thus ensuring decent performance on the test cases in most scenarios. Use ensembling to capture linear and simple as well non-linear complex relationships in the data. This can be done by using two different models and forming an ensemble of two.

There can be a countless number of ways in which you can ensemble different models. But these are some techniques that are mostly used:

**1.1.3.1 Bagging:** Bagging is also referred to as bootstrap aggregation. To understand bagging, we first need to understand bootstrapping. Bootstrapping is a sampling technique in which we choose ‘n’ observations or rows out of the original dataset of ‘n’ rows as well. But the key is that each row is selected with replacement from the original dataset so that each row is equally likely to be selected in each iteration

**1.1.3.2 Boosting:** Boosting is a sequential technique in which, the first algorithm is trained on the entire dataset and the subsequent algorithms are built by fitting the residuals of the first algorithm, thus giving higher weight to those observations that were poorly predicted by the previous model.

**1.1.3.3 Stacking:** In stacking multiple layers of machine learning models are placed one over another where each of the models passes their predictions to the model in the layer above it and the top layer model takes decisions based on the outputs of the models in layers below it.

* 1. **Problem Statement**

Diabetes, also known as chronic illness, is a group of metabolic diseases due to a high level of sugar in the blood over a long period. The risk factor and severity of diabetes can be reduced significantly if the precise early prediction is possible. How ensembling of classifiers can increase the accuracy and area under curve measures for predicting the diabetes?

**1.3 Objectives**

The objectives of this work are

1. To preprocess the dataset i.e., filling missing values, outlier rejection, feature selection.
2. To apply classifiers on dataset and tune parameters using Grid Search technique.
3. To predict diabetes with ensembling classifiers with more accuracy.

**1.4 Need of Present Study**

To know whether a person is diabetic or not , person has to diagnose in hospital with the help of sugar tolerance test. In previous days diabetes was diagnosed using several techniques like

**Random blood sugar test.** A blood sample will be taken at a random time. Regardless of when you last ate, a blood sugar level of 200 milligrams per deciliter (mg/dL) — 11.1 millimoles per liter (mmol/L) — or higher suggests diabetes.

**Fasting blood sugar test.** A blood sample will be taken after an overnight fast. A fasting blood sugar level less than 100 mg/dL (5.6 mmol/L) is normal. A fasting blood sugar level from 100 to 125 mg/dL (5.6 to 6.9 mmol/L) is considered prediabetes. If it's 126 mg/dL (7 mmol/L) or higher on two separate tests, you have diabetes.

**Oral glucose tolerance test.** For this test, you fast overnight, and the fasting blood sugar level is measured. Then you drink a sugary liquid, and blood sugar levels are tested periodically for the next two hours.

Now a days machine learning techniques are using to predict diabetes in advance. Early recognition of diabetes will help to cure diabetes easily. So Diabetes Prediction is required to predict whether person is diabetic or not by taking some details about them. So by taking some machine learning algorithms diabetes positive or negative can be predicted.

Although there are numerous frameworks already been published, in recent years, still, the improvement requires in the preciseness and robustness for diabetes prediction.

The present study is using ensembling of machine learning classifier to predict diabetes in more accurate way. **For predicting diabetes more accurately with some attributes collected from people help them to know about their diabetes diagnosis result in advance and help to them cure.**

In recent years, plenty of methods have been proposed and published for diabetes prediction. A ML based framework was proposed where authors implemented the Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Naive Bayes (NB), Gaussian Process Classification (GPC), Support Vector Machine (SVM), Artificial Neural Network (ANN), AdaBoost (AB), Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF) with different dimensionality reduction and cross-validation techniques.

Also performed extensive experiments on the outlier rejection and filling missing values for boosting the performance of the ML model, where they were able to obtain the highest possible AUC of 0.930. Other model employed three different ML classifiers such as DT, SVM, and NB to prognosticate the likelihood of diabetes with maximum accuracy and demonstrated that NB is the best performing model with the AUC of 0.819.

# 

**2. LITERATURE REVIEW**

There are many proposed frameworks for diabetes prediction using many different algorithms and different classifiers. Every individual framework proposed at that time have certain advantages over previous models and methodologies and certain drawbacks with present proposed robust frame works.

**2.1 Global Estimate of Diabetes**

**A.Misra et al. [3]** proposed that there has been a rapid escalation of type 2 diabetes (T2D) in developing countries, with varied prevalence according to rural vs urban habitat and degree of urbanization. Some ethnic groups (eg, South Asians, other Asians, and Africans), develop diabetes a decade earlier and at a lower body mass index than Whites, have prominent abdominal obesity, and accelerated the conversion from prediabetes to diabetes. The burden of complications, both macro- and microvascular, is substantial, but also varies according to populations. The syndemics of diabetes with HIV or tuberculosis are prevalent in many developing countries and predispose to each other. Screening for diabetes in large populations living in diverse habitats may not be cost-effective, but targeted high-risk screening may have a place.

The cost of diagnostic tests and scarcity of health manpower pose substantial hurdles in the diagnosis and monitoring of patients. Efforts for prevention remain rudimentary in most developing countries. The quality of care is largely poor; hence, a substantial number of patients do not achieve treatment goals. This is further amplified by a delay in seeking treatment, "fatalistic attitudes", high cost and non-availability of drugs and insulins. To counter these numerous challenges, a renewed political commitment and mandate for health promotion and disease prevention are urgently needed. Several low-cost innovative approaches have been trialed with encouraging outcomes, including training and deployment of non-medical allied health professionals and the use of mobile phones and telemedicine to deliver simple health messages for the prevention and management of T2D.

**Lancet. [11] proposed that** uncertainties persist about the magnitude of associations of diabetes mellitus and fasting glucose concentration with risk of coronary heart disease and major stroke subtypes. We aimed to quantify these associations for a wide range of circumstances. undertook a meta-analysis of individual records of diabetes, fasting blood glucose concentration, and other risk factors in people without initial vascular disease from studies in the Emerging Risk Factors Collaboration. We combined within-study regressions that were adjusted for age, sex, smoking, systolic blood pressure, and body-mass index to calculate hazard ratios (HRs) for vascular disease.

Analyses included data for 698 782 people (52 765 non-fatal or fatal vascular outcomes; 8.49 million person-years at risk) from 102 prospective studies. Adjusted HRs with diabetes were: 2.00 (95% CI 1.83-2.19) for coronary heart disease; 2.27 (1.95-2.65) for ischaemic stroke; 1.56 (1.19-2.05) for haemorrhagic stroke; 1.84 (1.59-2.13) for unclassified stroke; and 1.73 (1.51-1.98) for the aggregate of other vascular deaths. HRs did not change appreciably after further adjustment for lipid, inflammatory, or renal markers. Diabetes confers about a two-fold excess risk for a wide range of vascular diseases, independently from other conventional risk factors. In people without diabetes, fasting blood glucose concentration is modestly and non-linearly associated with risk of vascular disease.

**N. H. Cho et al. [17]** proposed that since the year 2000, IDF has been measuring the prevalence of diabetes nationally, regionally and globally. **Aim is** to produce estimates of the global burden of diabetes and its impact for 2017 and projections for 2045. A systematic literature review was conducted to identify published studies on the prevalence of diabetes, impaired glucose tolerance and hyperglycaemia in pregnancy in the period from 1990 to 2016. The highest quality studies on diabetes prevalence were selected for each country. A logistic regression model was used to generate age-specific prevalence estimates or each country. Estimates for countries without data were extrapolated from similar countries.

It was estimated that in 2017 there are 451 million (age 18-99 years) people with diabetes worldwide. These figures were expected to increase to 693 million) by 2045. It was estimated that almost half of all people (49.7%) living with diabetes are undiagnosed. Moreover, there was an estimated 374 million people with impaired glucose tolerance (IGT) and it was projected that almost 21.3 million live births to women were affected by some form of hyperglycaemia in pregnancy. In 2017, approximately 5 million deaths worldwide were attributable to diabetes in the 20-99 years age range. The global healthcare expenditure on people with diabetes was estimated to be USD 850 billion in 2017. The new estimates of diabetes prevalence, deaths attributable to diabetes and healthcare expenditure due to diabetes present a large social, financial and health system burden across the world

**P. Saeedi, et al. [22]** proposed to provide global estimates of diabetes prevalence for 2019 and projections for 2030 and 2045. A total of 255 high-quality data sources, published between 1990 and 2018 and representing 138 countries were identified. For countries without high quality in-country data, estimates were extrapolated from similar countries matched by economy, ethnicity, geography and language. Logistic regression was used to generate smoothed age-specific diabetes prevalence estimates (including previously undiagnosed diabetes) in adults aged 20-79 years.

The global diabetes prevalence in 2019 is estimated to be 9.3% (463 million people), rising to 10.2% (578 million) by 2030 and 10.9% (700 million) by 2045. The prevalence is higher in urban (10.8%) than rural (7.2%) areas, and in high-income (10.4%) than low-income countries (4.0%). One in two (50.1%) people living with diabetes do not know that they have diabetes. The global prevalence of impaired glucose tolerance is estimated to be 7.5% (374 million) in 2019 and projected to reach 8.0% (454 million) by 2030 and 8.6% (548 million) by 2045.Just under half a billion people are living with diabetes worldwide and the number is projected to increase by 25% in 2030 and 51% in 2045.

**2.2 Machine Learning Models**

**I. Jenhani, et al. [9]** proposed that deals withthe classification problem with imperfect data. More precisely, it extends standard decision trees to handle uncertainty in both building and classification procedures. Uncertainty here is represented by means of possibility distributions. The first part investigates the issue of building decision trees from data with uncertain class values by developing a non-specificity based gain ratio as the attribute selection measure which, in our case, is more appropriate than the standard gain ratio based on Shannon entropy. The proposed non-specificity based possibilistic decision tree (NS-PDT) approach is then extended by considering another kind of uncertainty inherent in the building procedure. The extended approach so-called non-specificity based possibilistic option decision tree (NS-PODT) offers a more flexible building procedure by allowing the selection of more than one attribute in each node. The second part addresses the classification phase. More specifically, it investigates the issue of predicting the class value of new instances presented with certain and/or uncertain attribute values. Finally, developed a possibilistic decision tree toolbox (PD2T) in order to show the feasibility of the proposed approach.

**L. Breiman [12]** proposed that random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges a.s. to a limit as the number of trees in the forest becomes large. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them. Using a random selection of features to split each node yields error rates that compare favorably to Adaboost , but are more robust with respect to noise. Internal estimates monitor error, strength, and correlation and these are used to show the response to increasing the number of features used in the splitting. Internal estimates are also used to measure variable importance. These ideas are also applicable to regression.

**R. Bansal, et al. [24]** proposed thatoutlier Detection is one of the major issues in Data Mining; finding outliers from a collection of patterns is a popular problem in the field of data mining. An outlier is that pattern which is dissimilar with respect to all the remaining patterns in the data set. Outlier detection is quiet familiar area of research in mining of data set. It is a quiet important task in various application domains. Earlier outliers considered as noisy data, has now become severe difficulty which has been discovered in various domains of research. The discovery of outlier is useful in detection of unpredicted and unidentified data, in certain areas like fraud detection of credit cards, calling cards, discovering computer intrusion and criminal behaviors etc. A number of surveys, research and review articles cover outlier detection techniques in great details. Here in this review paper, my effort is to take as one several techniques of outlier detection. By this attempt, there is gain a improved perceptive of various research on outlier detection and analysis for our well-being as well as for those who are the beginners in this field, so that they can easily pickup the links in details.

**D. Cousineau and S. Chartier [6]** proposedoutliers are observations or measures that are suspicious because they are much smaller or much larger than the vast majority of the observations. These observations are problematic because they may not be caused by the mental process under scrutiny or may not reflect the ability under examination. The problem is that a few outliers is sometimes enough to distort the group results (by altering the mean performance, by increasing variability, etc.). In this paper, various techniques aimed at detecting potential outliers are reviewed. These techniques are subdivided into two classes, the ones regarding univariate data and those addressing multivariate data. Within these two classes, we consider the cases where the population distribution is known to be normal, the population is not normal but known, or the population is unknown.

Outliers that are clearly the result of a spurious activity should be removed. However, in multivariate designs, doing so may result in removing too many participants to the point that the analysis can no longer be performed. Replacing the missing data with the mean of the remaining data in the corresponding cell. However, this procedure will tend to reduce the spread of the population, make the observed distribution more leptokurtic, and possibly increase the likelihood of a type-I error. A more elaborate technique, multiple imputations, involves replacing outliers (or missing data) with possible values.

**A. Hyvärinen and E. Oja. [1]** proposed fundamental problem in neural network research, as well as in many other disciplines, is finding a suitable representation of multivariate data, i.e. random vectors. For reasons of computational and conceptual simplicity, the representation is often sought as a linear transformation of the original data. In other words, each component of the representation is a linear combination of the original variables. Well-known linear transformation methods include principal component analysis, factor analysis, and projection pursuit. Independent component analysis (ICA) is a recently developed method in which the goal is to find a linear representation of non gaussian data so that the components are statistically independent, or as independent as possible. Such a representation seems to capture the essential structure of the data in many applications, including feature extraction and signal separation.

ICA is a very general-purpose statistical technique in which observed random data are linearly transformed into components that are maximally independent from each other, and simultaneously have “interesting” distributions. ICA can be formulated as the estimation of a latent variable model. The intuitive notion of maximum non gaussianity can be used to derive different objective functions whose optimization enables the estimation of the ICA model. Alternatively, one may use more classical notions like maximum likelihood estimation or minimization of mutual information to estimate ICA; somewhat surprisingly, these approaches are (approximatively) equivalent. A computationally very efficient method performing the actual estimation is given by the FastICA algorithm. Applications of ICA can be found in many different areas such as audio processing, biomedical signal processing, image processing, telecommunications, and econometrics.

**S. Arlot and A. Celisse [26]** designed to estimate the risk of an estimator or to perform model selection, cross-validation is a widespread strategy because of its simplicity and its apparent universality. Many results exist on the model selection performances of cross-validation procedures. This survey intends to relate these results to the most recent advances of model selection theory, with a particular emphasis on distinguishing empirical statements from rigorous theoretical results. As a conclusion, guidelines are provided for choosing the best cross-validation procedure according to the particular features of the problem in hand.

**P. Cunningham and S. J. Delany [20]** designed the most straightforward classifier in the arsenal or machine learning techniques is the Nearest Neighbour Classifier – classification is achieved by identifying the nearest neighbours to a query example and using those neighbours to determine the class of the query. The approach to classification is of particular importance today because issues of poor run-time performance is not such a problem these days with the computational power that is available. It presents an overview of techniques for Nearest Neighbour classification focusing on; mechanisms for assessing similarity (distance), computational issues in identifying nearest neighbours and mechanisms for reducing the dimension of the data.

**T. Chen and C. Guestrin [30]** designed tree boosting is a highly effective and widely used machine learning method. Model described a scalable end-to-end tree boosting system called XGBoost, which is used widely by data scientists to achieve state-of-the-art results on many machine learning challenges. Proposed a novel sparsity-aware algorithm for sparse data and weighted quantile sketch for approximate tree learning. More importantly, provided insights on cache access patterns, data compression and sharding to build a scalable tree boosting system. By combining these insights, XGBoost scales beyond billions of examples using far fewer resources than existing systems.

**A. S. Glas, et al. [4]** proposed that diagnostic testing can be used to discriminate subjects with a target disorder from subjects without it. Several indicators of diagnostic performance have been proposed, such as sensitivity and specificity. Using paired indicators can be a disadvantage in comparing the performance of competing tests, especially if one test does not outperform the other on both indicators. Here proposed the use of the odds ratio as a single indicator of diagnostic performance. The diagnostic odds ratio is closely linked to existing indicators, it facilitates formal meta-analysis of studies on diagnostic test performance, and it is derived from logistic models, which allow for the inclusion of additional variables to correct for heterogeneity. A disadvantage is the impossibility of weighing the true positive and false positive rate separately. In this article the application of the diagnostic odds ratio in test evaluation is illustrated.

**P. Ramachandran, et al. [21]** proposed the choice of activation functions in deep networks has a significant effect on the training dynamics and task performance. Currently, the most successful and widely-used activation function is the Rectified Linear Unit (ReLU). Although various hand-designed alternatives to ReLU have been proposed, none have managed to replace it due to inconsistent gains. Proposed work to leverage automatic search techniques to discover new activation functions. Using a combination of exhaustive and reinforcement learning-based search, discovered multiple novel activation functions.

Verified the effectiveness of the searches by conducting an empirical evaluation with the best discovered activation function. Our experiments show that the best discovered activation function, f(x)=x⋅sigmoid(βx), which we name Swish, tends to work better than ReLU on deeper models across a number of challenging datasets. For example, simply replacing ReLUs with Swish units improves top-1 classification accuracy on ImageNet by 0.9\% for Mobile NASNet-A and 0.6\% for Inception-ResNet-v2. The simplicity of Swish and its similarity to ReLU make it easy for practitioners to replace ReLUs with Swish units in any neural network.

**N. Srivastava, et al. [19]** designed deep neural nets with a large number of parameters are very powerful machine learning systems. However, overfitting is a serious problem in such networks. Large networks are also slow to use, making it difficult to deal with overfitting by combining the predictions of many different large neural nets at test time. Dropout is a technique for addressing this problem. The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents units from co-adapting too much. During training, dropout samples from an exponential number of different â thinnedâ networks. At test time, it is easy to approximate the effect of averaging the predictions of all these thinned networks by simply using a single unthinned network that has smaller weights. This significantly reduces overfitting and gives major improvements over other regularization methods. Dropout improves the performance of neural networks on supervised learning tasks in vision, speech recognition, document classification and computational biology, obtaining state-of-the-art results on many benchmark data sets.

**2.3 Neural Network Models**

**J. W. Smith, et al. [9]** designedneural networks or connectionist models for parallel processing are not new. However, a resurgence of interest in the past half decade has occurred. In part, this is related to a better understanding of what are now referred to as hidden nodes. These algorithms are considered to be of marked value in pattern recognition problems. Because of that, we tested the ability of an early neural network model, ADAP, to forecast the onset of diabetes mellitus in a high risk population of Pima Indians. The algorithm's performance was analyzed using standard measures for clinical tests: sensitivity, specificity, and a receiver operating characteristic curve. The crossover point for sensitivity and specificity is 0.76. We are currently further examining these methods by comparing the ADAP results with those obtained from logistic regression and linear perceptron models using precisely the same training and forecasting sets. A description of the algorithm is included.

Advantage is it is the first neural network model on diabetes prediction. Disadvantages are proposed model got less sensitivity and accuracy. It does not concentrate on pre processing.

**B. P. Tabaei and W. H. Herman [5]** designedneural networks have been trained to predict the subcellular location of proteins in prokaryotic or eukaryotic cells from their amino acid composition. For three possible subcellular locations in prokaryotic organisms a prediction accuracy of 81% can be achieved. Assigning a reliability index, 33% of the predictions can be made with an accuracy of 91%. For eukaryotic proteins (excluding plant sequences) an overall prediction accuracy of 66% for four locations was achieved, with 33% of the sequences being predicted with an accuracy of 82% or better. With the subcellular location restricting a protein's possible function, this method should be a useful tool for the systematic analysis of genome data and is available via a server on the world wide web.Disadvantage of model is it has very less accuracy. It has no preprocessing techniques.

**2.4 Genetic Programming**

**M. Pradhan and G. R. Bamnote [20]** designeddiabetes Mellitus is the one of the most serious health challenges. During the last 20 years the total number of diabetes patients has risen from 30 million to 230 million. It is a major health problem worldwide. So there is need of predictive model for early and accurate detection of diabetes. Diabetes disease diagnosis with proper interpretation of the diabetes data is an important classification problem. This research work proposes a Classifier for detection of Diabetes using Genetic programming (GP). Classification expression evaluation is used for creation of classifier. Reduced function pool of just arithmetic operators is used which allows for lesser validation and leniency during crossover and mutation.

Advantage is this model uses genetic programming which is new technique introduced. Genetic Programming (GP) is an evolutionary approach and is an alternative to traditional optimization or classification problems. The results of this approach can change drastically by experimenting with the crossover probability or by selection of different methods of population replacement and selection.It is very helpful when the developer does not have precise domain expertise as the algorithm possesses the ability to explore and learn from the domain. CPS logic eventually leads to enhanced performance of the proposed system.

Disadvantages are The effect of aging, diploidy, etc. on the performance and functionality of GP is not done in this model. For genetic programming there should be more need of medical and genetics domain knowledge. This is more difficult for other domain people to do work. This framework accuracy is less and doesnot find AUC to validate it. It doesn’t uses any data preprocessing techniques and doesn’t much concentrate on outliers.

**2.5 Single Machine Learning Algorithms**

**M. Maniruzzaman,et al. [15]** proposed diabetes is a silent killer. The main cause of this disease is the presence of excessive amounts of metabolites such as glucose. There were about 387 million diabetic people all over the world in 2014. The financial burden of this disease has been calculated to be about $13,700 per year. According to the World Health Organization (WHO), these figures will more than double by the year 2030. This cost will be reduced dramatically if someone can predict diabetes statistically on the basis of some covariates. Although several classification techniques are available, it is very difficult to classify diabetes. The main objectives of this paper are as follows: (i) Gaussian process classification (GPC), (ii) comparative classifier for diabetes data classification, (iii) data analysis using the cross-validation approach, (iv) interpretation of the data analysis and (v) benchmarking our method against others.

To classify diabetes, several classification techniques are used such as linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), and Naive Bayes (NB). However, most of the medical data show non-normality, non-linearity and inherent correlation structure. So in this paper we adapted Gaussian process (GP)-based classification technique using three kernels namely: linear, polynomial and radial basis kernel. We also investigate the performance of a GP-based classification technique in comparison to existing techniques such as LDA, QDA and NB. Performances are evaluated by using the accuracy (ACC), sensitivity (SE), specificity (SP), positive predictive value (PPV), negative predictive value (NPV) and receiver-operating characteristic (ROC) curves.

Pima Indian diabetes dataset is taken as part of the study. This consists of 768 patients, of which 268 patients are diabetic and 500 patients are controls. Our machine learning system shows the performance of GP-based model as: ACC 81.97%, SE 91.79%, SP 63.33%, PPV 84.91% and NPV 62.50% which are larger compared to other methods.

Disadvantages are This model uses median as measure of central tendency for data preprocessing. This model have less specificity i.e., it predicts the persons with out disease as diabetic persons.

**Q. Wang, et al. [23]** proposed diabetes mellitus is called a silent killer. It makes the body produce less insulin and causes increased blood sugar, which leads to many complications and affects the normal functioning of various organs, such as eyes, kidneys, and nerves. Although diabetes has attracted high attention in research, due to the existence of missing values and class imbalance in the data, the overall performance of diabetes classification using machine learning is relatively low. Proposed an effective Prediction algorithm for Diabetes Mellitus classification on Imbalanced data with Missing values (DMP\_MI). First, the missing values are compensated by the Naïve Bayes (NB) method for data normalization. Then, an adaptive synthetic sampling method (ADASYN) is adopted to reduce the influence of class imbalance on the prediction performance. Finally, a random forest (RF) classifier is used to generate predictions and evaluated using comprehensive set of evaluation indicators. Experiments performed on Pima Indians diabetes dataset from the University of California at Irvine, Irvine (UCI) Repository, have demonstrated the effectiveness and superiority of our proposed DMP\_MI.

Disadvantage is it has less sensitivity and just done one preprocessing step of filling missing values. No other preprocessing are done.

**S. P. Chatrati, et al. [28]** proposed that home health monitoring can facilitate patient monitoring remotely for diabetes and blood pressure patients. Early detection of hypertension and diabetes is extremely important, as these chronic diseases often result in life-threatening complications when found at a later stage. Work proposes a smart home health monitoring system that helps to analyze the patient’s blood pressure and glucose readings at home and notifies the healthcare provider in case of any abnormality detected. A combination of conditional decision-making and machine-learning approaches is used to predict hypertension and diabetes status, respectively. The goal is to predict the hypertension and diabetes status using the patient’s glucose and blood pressure readings. Using supervised machine learning classification algorithms, herein a system is trained to predict the patient’s diabetes and hypertension status. After analyzing all the classification algorithms, support vector machine classification algorithm was found to be most accurate and thus chosen to train the model. This proposed work develops an application for a home health monitoring system with a user-friendly easy-to-use graphical user interface to diagnose blood pressure and diabetes status of patients along with sending categorized alerts and real-time notifications to their registered physician or clinic all from home.

Advantage is it is the first model that implemented a user friendly interface for predicting diabetes. Disadvantages are used only 2 attributes to predict diabetes. Not used any preprocessing techniques.

**2.6 Comparing different Machine Learning Algorithms**

**D. Sisodia and D. S. Sisodia [7]** proposed that diabetes is considered as one of the deadliest and chronic diseases which causes an increase in blood sugar. Many complications occur if diabetes remains untreated and unidentified. The tedious identifying process results in visiting of a patient to a diagnostic center and consulting doctor. But the rise in machine learning approaches solves this critical problem. The motive of this study is to design a model which can prognosticate the likelihood of diabetes in patients with maximum accuracy. Therefore three machine learning classification algorithms namely Decision Tree, SVM and Naive Bayes are used in this experiment to detect diabetes at an early stage. Experiments are performed on Pima Indians Diabetes Database (PIDD) which is sourced from UCI machine learning repository. The performances of all the three algorithms are evaluated on various measures like Precision, Accuracy, F-Measure, and Recall. Accuracy is measured over correctly and incorrectly classified instances. Results obtained show Naive Bayes outperforms with the highest accuracy of 76.30% comparatively other algorithms. These results are verified using Receiver Operating Characteristic (ROC) curves in a proper and systematic manner.

Advantages are Model gives best accuracy than all other proposed models. Naive Bayes outperforms with the highest accuracy of 76.30% comparatively other algorithms. This helped for next proposed models to choose Naïve Bayes as one of the Classifier.

Disadvantages are Model provides best accuracy but not good AUC(0.819). This model uses only three Classifiers and check their performances in separate .It didn’t use any Ensembling techinques for better prediction. It doesn’t concentrate on data preprocessing and feature selection.

**N. Nai-arun and R. Moungmai [18]** proposed model applied a use of algorithms to classify the risk of diabetes mellitus. Four well known classification models that are Decision Tree, Artificial Neural Networks, Logistic Regression and Naive Bayes were first examined. Then, Bagging and Boosting techniques were investigated for improving the robustness of such models. Additionally, Random Forest was not ignored to evaluate in the study. Findings suggest that the best performance of disease risk classification is Random Forest algorithm. Therefore, its model was used to create a web application for predicting a class of the diabetes risk.

Advantages are the best algorithm where gives the highest accuracy or ROC Curve value is selected for creating a web application of diabetes risk prediction. A real dataset is used where data collected from 30,122 people who were collected from 26 Primary Care Units in Sawanpracharak Regional Hospital during 2012 – 2013.

Disadvantages are the proposed framework have less AUC(0.912) than previous proposed models. This model doesn’t much concentrate on data preprocessing and feature selection.

**H. Kaur and V. Kumari [8]** proposed that early detection of diabetes is very important to maintain a healthy life. This disease is a reason of global concern as the cases of diabetes are rising rapidly. Machine learning (ML) is a computational method for automatic learning from experience and improves the performance to make more accurate predictions. In the current research we have utilized machine learning technique in Pima Indian diabetes dataset to develop trends and detect patterns with risk factors using R data manipulation tool. To classify the patients into diabetic and non-diabetic we have developed and analyzed five different predictive models using R data manipulation tool. For this purpose we used supervised machine learning algorithms namely linear kernel support vector machine (SVM-linear), radial basis function (RBF) kernel support vector machine, k-nearest neighbour (k-NN), artificial neural network (ANN) and multifactor dimensionality reduction (MDR).

Such a high value of AUC

indicates that both SVM- linear and k-NN are optimal classiﬁers

for diabetic dataset. So, from above studies, it can be said that on

the basis of all the parameters linear kernel support vector

machine (SVM-linear) and k-NN are two best models to ﬁnd that

whether patient is diabetic or not.

This work also suggests that Boruta wrapper algorithm can be

used for feature selection. The experimental results indicated that

using the Boruta wrapper features selection algorithm is better

than choosing the attributes manually with less medical domain

knowledge. Thus with a limited number of parameters, through

the Boruta feature selection algorithm we have achieved higher

accuracy and precisio

Such a high value of AUC

indicates that both SVM- linear and k-NN are optimal classiﬁers

for diabetic dataset. So, from above studies, it can be said that on

the basis of all the parameters linear kernel support vector

machine (SVM-linear) and k-NN are two best models to ﬁnd that

whether patient is diabetic or not.

This work also suggests that Boruta wrapper algorithm can be

used for feature selection. The experimental results indicated that

using the Boruta wrapper features selection algorithm is better

than choosing the attributes manually with less medical domain

knowledge. Thus with a limited number of parameters, through

the Boruta feature selection algorithm we have achieved higher

accuracy and precisio

Such a high value of AUC

indicates that both SVM- linear and k-NN are optimal classiﬁers

for diabetic dataset. So, from above studies, it can be said that on

the basis of all the parameters linear kernel support vector

machine (SVM-linear) and k-NN are two best models to ﬁnd that

whether patient is diabetic or not.

This work also suggests that Boruta wrapper algorithm can be

used for feature selection. The experimental results indicated that

using the Boruta wrapper features selection algorithm is better

than choosing the attributes manually with less medical domain

knowledge. Thus with a limited number of parameters, through

the Boruta feature selection algorithm we have achieved higher

accuracy and precisio

Such a high value of AUC

indicates that both SVM- linear and k-NN are optimal classiﬁers

for diabetic dataset. So, from above studies, it can be said that on

the basis of all the parameters linear kernel support vector

machine (SVM-linear) and k-NN are two best models to ﬁnd that

whether patient is diabetic or not.

This work also suggests that Boruta wrapper algorithm can be

used for feature selection. The experimental results indicated that

using the Boruta wrapper features selection algorithm is better

than choosing the attributes manually with less medical domain

knowledge. Thus with a limited number of parameters, through

the Boruta feature selection algorithm we have achieved higher

accuracy and precisio

Such a high value of AUC

indicates that both SVM- linear and k-NN are optimal classiﬁers

for diabetic dataset. So, from above studies, it can be said that on

the basis of all the parameters linear kernel support vector

machine (SVM-linear) and k-NN are two best models to ﬁnd that

whether patient is diabetic or not.

This work also suggests that B

Such a high value of AUC

indicates that both SVM- linear and k-NN are optimal classiﬁers

for diabetic dataset. So, from above studies, it can be said that on

the basis of all the parameters linear kernel support vector

machine (SVM-linear) and k-NN are two best models to ﬁnd that

whether patient is diabetic or not.

This work also suggests that B

Such a high value of AUC

indicates that both SVM- linear and k-NN are optimal classiﬁers

for diabetic dataset. So, from above studies, it can be said that on

the basis of all the parameters linear kernel support vector

machine (SVM-linear) and k-NN are two best models to ﬁnd that

whether patient is diabetic or not.

This work also suggests that B

Such a high value of AUC indicates that both SVM- linear and k-NN are optimal classiﬁers for diabetic dataset. So, from above studies, it can be said that on the basis of all the parameters linear kernel support vector machine (SVM-linear) and k-NN are two best models to ﬁnd thatwhether patient is diabetic or not**.**

This work also suggests that Boruta wrapper algorithm can be

used for feature selection. The experimental results indicated that

using the Boruta wrapper features selection algorithm is better

than choosing the attributes manually with less medical domain

knowledge. Thus with a limited number of parameters, through

the Boruta feature selection algorithm we have achieved higher

accuracy and precision

This work also suggests that Boruta wrapper algorithm can be

used for feature selection. The experimental results indicated that

using the Boruta wrapper features selection algorithm is better

than choosing the attributes manually with less medical domain

knowledge. Thus with a limited number of parameters, through

the Boruta feature selection algorithm we have achieved higher

accuracy and precision

This work also suggests that Boruta wrapper algorithm can be

used for feature selection. The experimental results indicated that

using the Boruta wrapper features selection algorithm is better

than choosing the attributes manually with less medical domain

knowledge. Thus with a limited number of parameters, through

the Boruta feature selection algorithm we have achieved higher

accuracy and precision

This work also suggests that Boruta wrapper algorithm can be used for feature selection. The experimental results indicated that using the Boruta wrapper features selection algorithm is better than choosing the attributes manually with less medical domain knowledge. Thus with a limited number of parameters, through the Boruta feature selection algorithm we have achieved higher accuracy and precision.

**2.7 Models with Preprocessing**

**R. Vaishali, et al. [25]** proposed that diabetes Mellitus is a dreadful disease characterized by increased levels of glucose in the blood, termed as the condition of hyperglycemia. As this disease is prominent among the tropical countries like India, an intense research is being carried out to deliver a machine learning model that could learn from previous patient records in order to deliver smart diagnosis. This research work aims to improve the accuracy of existing diagnostic methods for the prediction of Type 2 Diabetes with machine learning algorithms. The proposed algorithm selects the essential features from the Pima Indians Diabetes Dataset with Goldberg's Genetic algorithm in the pre-processing stage and a Multi Objective Evolutionary Fuzzy Classifier is applied on the dataset. This algorithm works on the principle of maximum classifier rate and minimum rules. As a result of feature selection with GA the number of features is reduced to 4 from 8 and the classifier rate is improved to 83.0435 % with NSGA II in training rate of 70% and 30% testing.

Advantage is it used preprocessing technique for feature selection. Disadvantage is it selected and used only 4 features.Accuracy and specificity are less.

**M. Maniruzzaman, et al. [14]** objective of this frame work to develop an optimized and robust machine learning (ML) system under the assumption that missing values or outliers if replaced by a median configuration will yield higher risk stratification accuracy. This ML-based risk stratification is designed, optimized and evaluated, where: the features are extracted and optimized from the six feature selection techniques (random forest, logistic regression, mutual information, principal component analysis, analysis of variance, and Fisher discriminant ratio).

Advantages are The random forest feature selection and random forest classification technique yields an accuracy, sensitivity, specificity, positive predictive value, negative predictive value and area under the curve as: 92.26%, 95.96%, 79.72%, 91.14%, 91.20%, and 0.93, respectively. This is an improvement of 10% over previously developed techniques published. RF-based model showed the best performance when outliers are replaced by median values. Disadvantages are This model have less accuracy and less specificity and AUC. This model uses median to fill outliers and missing values which work well for only Random Forest Classifier. Diabetes Prediction Using Ensembling of Different Machine Learning Classifiers framework uses mean as the best central tendency to fill missing values and outliers for best AUC value.

**2.8 Ensembling Models**

**S. Perveen, et al. [29]** proposed that diabetes Mellitus is one of the major health challenges all over the world. The prevalence of diabetes is increasing at a fast pace, deteriorating human, economic and social fabric. Prevention and prediction of diabetes mellitus is increasingly gaining interest in healthcare community. Although several clinical decision support systems have been proposed that incorporate several data mining techniques for diabetes prediction and course of progression. These conventional systems are typically based either just on a single classifier or a plain combination thereof. Recently extensive endeavors are being made for improving the accuracy of such systems using ensemble classifiers. This study follows the adaboost and bagging ensemble techniques using J48 (c4.5) decision tree as a base learner along with standalone data mining technique J48 to classify patients with diabetes mellitus using diabetes risk factors. This classification is done across three different ordinal adults groups in Canadian Primary Care Sentinel Surveillance network. Experimental result shows that, overall performance of adaboost ensemble method is better than bagging as well as standalone J48 decision tree.

Disadvantages are It doesn’t work much on data preprocessing techniques and feature selection. It has less sensitivity and less AUC.

**L. Li implemented [13]** proposed that diabetes is a worldwide public health challenge with a yearly increasing incidence. Many approaches using different machine learning classifiers have been developed for automatic diagnosis of diabetes. However, they mostly rely on a single classifier or a hybrid model to make the diagnosis decision, which might be weaker than a voted decision of multiple classifiers. In this study, we present an approach that combines three classifiers (i.e. Support vector machine, artificial neural network, and naïve bayes) to diagnose diabetes. The approach can adjust each classifier's weight based on their ability and history of making correct predictions. A rule that mixes majority voting and weights of classifiers was proposed and applied for the final diagnosis decision. The Pima Indians diabetes data set (268 diabetes patients and 500 normal subjects) was used in the work. A wrapper method was adopted to select features for classification. An experimental comparison of our method with other voting strategies and each single classifier used in our study demonstrated that our approach performed better in sensitivity.

Disadvantages are It has less sensitivity of 0.583. So , It doesn’t correctly identify the persons with disease. It doesn’t use AUC as the validation parameter. So, This model don’t validated correctly and can’t find that predicted value is correct or not.

**A. K. Dewangan and P. Agrawal [2]** Diabetes-Mellitus refers to the metabolic disorder that happens from misfunction in insulin secretion and action. It is characterized by hyperglycemia. The persistent hyperglycemia of diabetes leads to damage, malfunction and failure of different organs such as kidneys, eyes, nerves, blood vessels and heart. In the past decades several techniques have been implemented for the detection of diabetes. The diagnosis of diabetes is very important now a days using various types of techniques. Here, there are various techniques, their classification and implementation using various types of software tools and techniques. The diagnosis of diabetes can be done using Artificial Neural Network, K-fold cross validation and classification, Vector support machine, K-nearest neighbour method, Data Mining Algorithm, etc. Using these techniques, we attempt to make an ensemble model by combining two techniques: Bayesian classification and Multilayer Perceptron for the accuracy, sensitivity and specificity measures of diagnosis of diabetes-mellitus.

Disadvantages are it doesn’t concentrate on preprocessing techniques and doesn’t much work about outlier rejection and missing values. It has less value of sensitivity (0.641). For feature selection this model selects the required features manually.

**S. Bashir, et al. [27]**  proposed that accuracy plays a vital role in the medical field as it concerns with the life of an individual. Extensive research has been conducted on [disease classification](https://www.sciencedirect.com/topics/medicine-and-dentistry/nosology) and prediction using [machine learning techniques](https://www.sciencedirect.com/topics/computer-science/machine-learning-technique). However, there is no agreement on which classifier produces the best results. A specific classifier may be better than others for a specific dataset, but another classifier could perform better for some other dataset. Ensemble of classifiers has been proved to be an effective way to improve [classification accuracy](https://www.sciencedirect.com/topics/computer-science/classification-accuracy). In this research we present an ensemble framework with multi-layer classification using enhanced bagging and optimized weighting. The proposed model called “HM-BagMoov” overcomes the limitations of conventional [performance bottlenecks](https://www.sciencedirect.com/topics/computer-science/performance-bottleneck) by utilizing an ensemble of seven heterogeneous classifiers. The framework is evaluated on five different heart disease datasets, four breast cancer datasets, two diabetes datasets, two liver disease datasets and one hepatitis dataset obtained from public repositories. The analysis of the results show that ensemble framework achieved the highest accuracy, sensitivity and F-Measure when compared with [individual classifiers](https://www.sciencedirect.com/topics/computer-science/individual-classifier) for all the diseases. In addition to this, the ensemble framework also achieved the highest accuracy when compared with the state of the art techniques. An application named “IntelliHealth” is also developed based on proposed model that may be used by hospitals/doctors for diagnostic advice.

Advantages are Extensive research has been conducted on diabetes prediction. An optimal combination of classifiers is presented with multi-layer classification. The emsemble approach uses bagging with multi-objective optimized weighted. Comparision with existing techniques show superiority of ensemble.

Disadvantages are it has less accuracy and specificity. It does not concentrate on preprocessing.

From all these related works each model used different approach. Each have their own advantage and disadvantage. Some models used only one algorithm. Some used naïve approaches without optimization. Some models does not concentrate on preprocessing. These models have less accuracy, sensitivity and specificity. To overcome all these draw backs and improve prediction accuracy by doing preprocessing and ensembling machine learning classifiers using soft voting technique.

**3. SYSTEM ANALYSIS**

**3.1 Requirement Specification**

A Software Requirement Specifications (SRS) – a Requirements Specification for a Software System is a finished depiction of the conduct of a framework to be created. It incorporates a lot of utilization cases that depict every one of the associations the clients will have with the product. Not with standing use cases, the SRS additionally contains non-useful necessities. Non-practical necessities are prerequisites which force limitations on the plan or usage, (for example, operational efficiency necessities, quality gauges, or structure imperatives).

* Business necessities portray in business terms what must be conveyed or achieved to offer some incentive.
* Product prerequisites portray properties of a framework or item (which could be one of a few different ways to achieve a lot of business necessities.
* Process prerequisites portray exercises performed by the creating association. For example, process necessities could determine explicit approaches that must be pursued, and requirements that the association must comply.

Item and procedure prerequisites are firmly connected. Procedure prerequisites frequently determine the exercises that will be performed to fulfill an item necessity. For instance, a most extreme advancement cost necessity (a procedure prerequisite) might be forced to help accomplish a greatest deals value necessity (an item necessity); a prerequisite that the item be viable (a Product prerequisite) frequently is tended to by forcing necessities to pursue specific improvement styles.

Requirements are:

1. Detecting a person is diabetic or not.
2. Better performance with best Accuracy.
3. Accelerate Training and Testing.

**3.1.1 Functional Requirements**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation. The various types of outputs in general are:

* External Outputs, whose destination is outside the organization.
* Internal Outputs whose destination is within organization.
* User’s main interface with the computer.
* Operational outputs whose use is purely within the computer department.
* Interface outputs, which involve the user in communicating directly.
* Understanding user’s preferences, expertise level and his business requirements.

Predicting Diabetes

**Input:** It takes all 8 attributes(Pregnancies, Age, BMI, Skin Thickness etc…) as an input from the person.

**Output:** Predict whether a person is diabetic or not.

**3.1.2 Non Functional Requirements**

Non-Functional Requirements specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system. Example of nonfunctional requirement, “how fast does the website load?” Failing to meet non-functional requirements can result in systems that fail to satisfy user needs. Non-functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are greater than 10000. Description of non-functional requirements is just as critical as a functional requirement.

* Usability requirement
* Serviceability requirement
* Manageability requirement
* Recoverability requirement
* Security requirement
* Data Integrity requirement
* Capacity requirement
* Availability requirement
* Scalability requirement
* Interoperability requirement
* Reliability requirement
* Maintainability requirement
* Regulatory requirement
* Environmental requirement
  + - 1. **Execution qualities:**

Efficiency: The state or quality of being efficient, i.e., able to accomplish something with the least waste of time and effort; competency in performance.

* + - 1. **Evolution qualities:**

Testability: means by which the presence, quality, or genuineness of anything is determined.

Extensibility: to enlarge the scope of, or make more comprehensive, as operations, influence etc.

Scalability: The ability of something, especially a computer system, to adapt to the increased demands.

**3.1.3 User requirements**

* Execution time should be fast
* More Accurate
* User Friendly

**3.1.4 Software requirements**

* O/S : Windows
* Language : Python3

**3.1.5 Hardware requirements**

* SYSTEM : Pentium Dual Core
* HARD DISK : 120 GB
* MONITOR
* MOUSE
* RAM : 8 GB or more

**3.2 UML Diagrams for project work**

UML is an acronym that stands for Unified Modeling Language. Simply put, UML is a modern approach to modeling and documenting software. In fact, it’s one of the most popular business process modeling techniques.

It is based on diagrammatic representations of software components. As the old proverb says: “a picture is worth a thousand words”. By using visual representations, we are able to better understand possible flaws or errors in software or business processes.

The elements are like components which can be associated in different ways to make a complete UML picture, which is known as diagram. Thus, it is very important to understand the different diagrams to implement the knowledge in real-life systems.

Any complex system is best understood by making some kind of diagrams or pictures. These diagrams have a better impact on our understanding. If we look around, we will realize that the diagrams are not a new concept but it is used widely in different forms in different industries.

Prepare UML diagrams to understand the system in a better and simple way. A single diagram is not enough to cover all the aspects of the system. UML defines various kinds of diagrams to cover most of the aspects of a system. UML was created as a result of the chaos revolving around software development and documentation. In the 1990s, there were several different ways to represent and document software systems. The need arose for a more unified way to visually represent those systems and as a result, in 1994-1996, the UML was developed by three software engineers working at Rational Software.

Mainly, UML has been used as a general-purpose modeling language in the field of software engineering. However, it has now found its way into the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts.

They provide both a more standardized way of modeling workflows as well as a wider range of features to improve readability and efficiency. Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system. Since all the needs of a system typically cannot be covered in one use case, it is usual to have a collection of use cases. Together this use case collection specifies all the ways the system. An association provides a pathway for communication. The communication can be between use cases, actors, classes or interfaces. Associations are the most general of all relationships and consequentially the most semantically weak. If two objects are usually considered independently, the relationship is an association. They provide both a more standardized way of modeling workflows as well as a wider range of features to improve readability and efficiency. Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system. Since all the needs of a system typically cannot be covered in one use case, it is usual to have a collection of use cases.

By default, the association tool on the toolbox is unidirectional and drawn on a diagram with a single arrow at one end of the association. The end with the arrow indicates who or what is receiving the communication. A dependency is a relationship between two model elements in which a change to one model element will affect the other model element. Typically, on class diagrams, a dependency relationship indicates that the operations of the client invoke operations of the supplier. The work flow in this case begins from importing the dataset by the developer and then replacing missing values with mean value of corresponding column, model building, validating that model by generating a confusion matrix and finally predicting the test sample class label. Transitions are used to show the passing of the flow of control from activity.

The various UML diagrams are

* Usecase diagram
* Activity diagram
* Sequence diagram
* Colloboration diagram
* Object diagram
* State chart diagram
* Class diagram
* Component diagram
* Deployment diagram

**3.2.1 Usecase Diagram:**

A use case diagram is a graph of actors, a set of use cases enclosed by a system boundary, communication (participation) associations between the actors and users and generalization among use cases. The use case model defines the outside (actors) and inside (use case) of the system’s behavior. Actors are not part of the system. Actors represent anyone or anything that interacts with (input to or receive output from) the system. Use-case diagrams can be used during analysis to capture the system requirements and to understand how the system should work. During the design phase, you can use use-case diagrams to specify the behavior of the system as implemented. Use case is a sequence of transactions performed by a system that yields a measurable result of values for a particular actor. The use cases are all the ways the system may be used.

Use case is a list of actions or event steps, typically defining the interactions between a role (known as an actor) and a system, to achieve a goal. In case of the use case diagram developer and the end user are the actors. Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system. Since all the needs of a system typically cannot be covered in one use case, it is usual to have a collection of use cases. Together this use case collection specifies.

**Figure 3.2.1.1: Usecase Diagram**

An include relationship is a stereotyped relationship that connects a base use case to an inclusion use case. An include relationship specifies how behavior in the inclusion use case is used by the base use case. An extend relationship is a stereotyped relationship that specifies how the functionality of one use case can be inserted into the functionality of another use case. Extend relationships between uses cases are modeled as dependencies by using the Extend stereotype. The different use cases are import dataset for importing datasets, preprocessing, model building, validation and prediction.

**3.2.2 Activity Diagram:**

An Activity diagram is a variation of a special case of a state machine, in which the states are activities representing the performance of operations and the transitions are triggered by the completion of the operations. The purpose of Activity diagram is to provide a view of flows and what is going on inside a use case or among several classes. Activity diagrams contain activities, transitions between the activities, decision points, and synchronization bars.

An activity represents the performance of some behavior in the workflow. In the UML, activities are represented as rectangles with rounded edges, transitions are drawn as directed arrows, decision points are shown as diamonds, and synchronization bars are drawn as thick horizontal or vertical bars as shown in the following. The activity icon appears as a rectangle with rounded ends with a name and a component for actions.

The work flow in this case begins from importing the configuration files and database and then preprocessing the sentence to parse. After parsing sentence is split into sub queries and then estimating the attributes. Finally query is formed. Transitions are used to show the passing of the flow of control from activity to activity. They are typically triggered by the completion of the behavior in the originating activity.



**Figure 3.2.2.1: Activity Diagram**

Transition connects activities with other model elements and object flows connect activities with objects. They are typically triggered by the completion of the behavior in the originating activity.

Swim lanes may be used to partition an activity diagram. This typically is done to show what person or organization is responsible for the activities contained in the swim lane. Swim lanes are helpful when modeling a business workflow because they can represent organizational units or roles within a business model. Swim lanes are very similar to an object because they provide a way to tell who is performing a certain role. Swim lanes only appear on activity diagrams. When a swim lane is dragged onto an activity diagram, it becomes a swim lane view. Swim lanes appear as small icons in the browse while a swim lane views appear between the thin, vertical lines with a header that can be renamed and relocated. An activity represents the performance of some behavior in the work flow. In the UML, activities are rep- resented as rectangles with rounded edges, transitions are drawn as directed arrows, decision 17 points are shown as diamonds, and synchronization bars are drawn as thick horizontal or vertical bars as shown in the following. The activity icon appears as a rectangle with rounded ends with a name and a component for actions.

**3.2.3 Sequence Diagram**

A sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called as event diagrams.



**Figure 3.2.3.1: Sequence Diagram**

**3.2.4 Collaboration Diagram**

A collaboration diagram shows that the order of messages that implement an operation or a transaction. Collaboration diagrams show objects, their links and their messages. They can also contain simple class instances and class utility instances. Each collaboration diagram provides a view of the interactions or structural relationships that occur between objects and object like entities in the current model. Collaboration diagrams and sequence diagrams are called interaction diagrams. A collaboration diagram shows that the order of messages that implement an operation or a transaction. Collaboration diagrams show objects, their links, and their messages. They can also contain simple class instances and class utility instances. A collaboration diagram shows that the order of messages that implement an operation or a transaction. Collaboration diagrams show objects, their links, and their messages. They can also contain simple class instances and class utility instances. Each collaboration diagram provides a view of the interactions or structural relationships that occur between objects and object like entities in the current model.

The second interaction diagram is the collaboration diagram. It shows the object organization as seen in the following diagram. In the collaboration diagram, the method call sequence is indicated by some numbering technique. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. Method calls are similar to that of a sequence diagram. However, difference being the sequence diagram does not describe the object organization, whereas the collaboration diagram shows the object organization. To choose between these two diagrams, emphasis is placed on the type of requirement. If the time sequence is important, then the sequence diagram is used. If organization is required, then collaboration diagram is used. Interaction diagrams are used to describe the dynamic nature of a system. Now, we will look into the practical scenarios where these diagrams are used. To understand the practical application, we need to understand the basic nature of sequence and collaboration diagram.

The main purpose of both the diagrams are similar as they are used to capture the dynamic behavior of a system. However, the specific purpose is more important to clarify and understand.

Sequence diagrams are used to capture the order of messages flowing from one object to another. Collaboration diagrams are used to describe the structural organization of the objects taking part in the interaction. A single diagram is not sufficient to describe the dynamic aspect of an entire system, so a set of diagrams are used to capture it as a whole. Interaction diagrams are used when we want to understand the message flow and the structural organization. Message flow means the sequence of control flow from one object to another. Structural organization means the visual organization of the elements in a system.

**Figure 3.2.4.1. Collaboration Diagram**

**3.2.5 Class Diagram:**

Class diagrams contain icons representing classes, interfaces and the relationships. You can create one or more class diagrams to represent the classes at the top level of the current model; such class diagrams are themselves contained by the top level of the current model. You can also create one or more class diagrams to represent classes contained by each package in your model; such class diagrams are themselves contained by the package enclosing the classes they represent; the icons representing logical packages and classes in class diagrams.

* Class diagrams are created to provide a picture or view of some or all of the classes in the model.
* The main class diagram in the logical view of the model is typically a picture of the packages in the’ system. Each package also has its own main class diagram, which typically displays the “public” classes of the package.

A class diagram is a picture for describing generic descriptions of possible systems. Class diagrams and collaboration diagrams are alternate representations of object models.

A Class is a description of a group of objects with common properties (attributes) common behavior (operations), common relationships too their objects, and common semantics. Thus, a class is a template to create objects. Each object is

an instance of some class and objects cannot be instances of more than one class. In the UML, classes are represented as compartmentalized rectangles.

* The top compartment contains the name of the class.
* The middle compartment contains the structure of the class (attributes).
* The bottom compartment contains the behaviour of the class (operations).

**Figure 3.2.5.1: Class Diagram**

**3.2.6 State Chart Diagram**

Use cases and scenarios provide a way to describe system behavior; in the form of interaction between objects in the system. Sometime it is necessary to consider inside behavior of an object. A state chart diagram shows the states of a single objects, the events or messages that cause a transition from one state to an other and the actions that result from a state change. As activity diagram, state chart diagram also contains special symbols for start state and stop state.

State chart diagram cannot be created for every class in the system, it is only for those lass objects with significant behavior. State transition: A state transition indicates that an object in the source state will perform certain specified actions and enter the destination state when a specified event occurs or when certain conditions are satisfied.

A state transition is a relationship between two states, two activities, or between an activity and a state. We can show one or more state transitions from a state as long as each transition is unique. Transitions originating from a state cannot have the same event, unless there are conditions on the event. Provide a label for each state transition with the name of at least one event that causes the state transition. You do not have to use unique labels for state transitions because the same event can cause a transition to many different states or activities.

**3.2.7 Component Diagram**

Component Diagrams show the dependencies between software components in the system. The nature of these dependencies will depend on the language or languages used for the development and may exist at compile-time or at runtime.

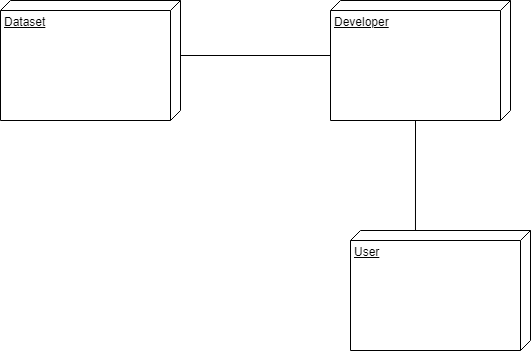
In a large project there will be many files that makeup the system. These files will have dependencies on one another. The nature of these dependencies will depend on the language or languages used for the development and may exist at compile-time, at link-time or at run-time. There are also dependencies between source code files and the executable files or bytecode files that are derived from them by compilation. Component diagrams are one of the two types of implementation diagram in UML. Component diagrams show these dependencies between software components in the system. Stereotypes can be used to show dependencies that are specific to particular languages also.

 A component diagram shows the allocation of classes and objects to components in the physical design of a system. A component diagram may represent all or part of the component architecture of a system along with dependency relationships. The dependency relationship indicates that one entity in a component diagram uses the services or facilities of another.

**Figure 3.2.7.1: Component Diagram**

**3.2.8 Deployment Diagram:**

The second type of implementation diagram provided by UML is the deployment diagram. Deployment diagrams are used to show the configuration of run time processing elements and the software components and processes that are located on them. Deployment diagrams are made up of nodes and communication associations.

 Nodes are typically used to show computers and the communication associations show the network and protocols that are used to communicate between nodes. Nodes can be used to show other processing resources such as people or mechanical resources. Nodes are drawn as 3D views of cubes or rectangular prisms, and the following figure shows a simplest deployment diagram where the nodes connected by communication associations.

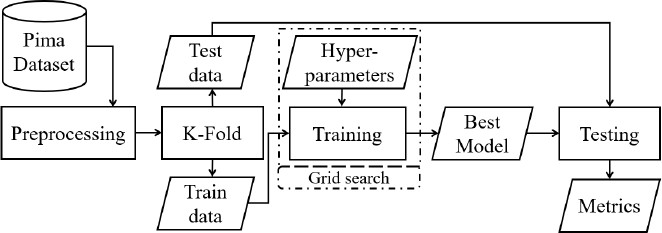
**Figure 3.2.8.1: Deployment Diagram**

**4. SYSTEM DESIGN**

* 1. **Architecture**

Diabetes Prediction Using Ensembling of Different Machine Learning Classifiers model propose a new pipeline for diabetes prediction from the PIMA Indians Diabetes dataset.

Preprocessing, in the proposed pipeline, is the heart of achieving the state-of-the-art result, which consists of outlier rejection, filling missing values, data standardization, feature selection, and K-fold cross-validation.

 This proposed framework consider the mean value in the missing position of attribute rather than median value, as it has a more central tendency toward the mean of that attribute distribution. The folding of the dataset for cross-fold validation is performed carefully to preserve the percentage of class proportion, as same as in the original dataset.

**Figure 4.1.1: The proposed block diagram of a robust and automatic diabetes prediction.**

Different ML classifiers (k-nearest Neighbor (k-NN), RF, DT, NB, AB, and XGBoost (XB)) and MLP were implemented in the proposed pipeline. This framework apply the grid search technique for selecting the number of hidden layers, number of neurons in each hidden layer, activation function, neuron initializer, batch size, learning rate, epoch, percentage of dropped neurons, loss function, an optimizer of MLP and hyperparameters of ML models. Extensive experiments are performed on different combinations of preprocessing and ML classifiers for maximizing the AUC of diabetes prediction under the same experimental conditions and dataset.

The best ML classifier is then set as a baseline model to evaluate our proposed classifier quantitatively for the prediction of diabetes precisely. Moreover, the proposed model is an ensembling classifier by the combination of the ML models for boosting the diabetes prediction.

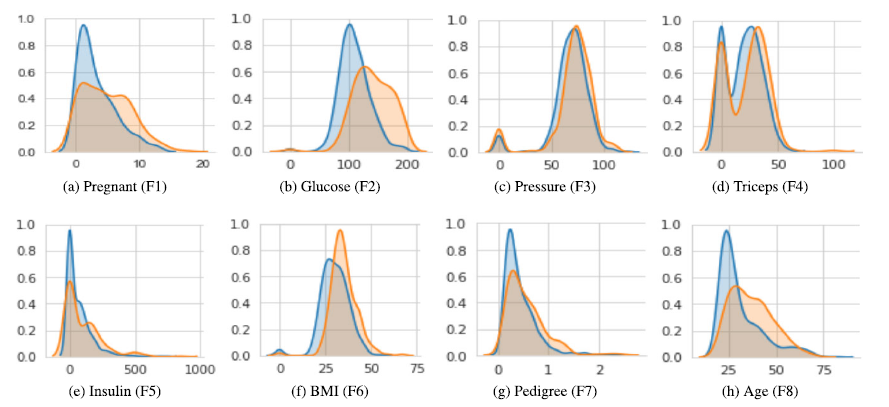
To ensemble the ML models, soft weighted voting is employed, where the weight for the individual model was estimated from the respective AUC.

The **AUC** of the ML model is chosen as the weight of that model for voting ensembling rather than accuracy since AUC is unbiased to the class distribution.

Extensive experiments on different combinations of the ML models are accomplished for searching the best ensemble classifier where the best performing preprocessing from the previous experiments is employed.

**5. IMPLEMENTATION**

* 1. **Modules**

 The prediction of diabetes is a challenging task, as the distribution of classes for all attributes is not linearly separable as depicted in below Fig.3.

**Figure 5.1.1: The population distribution of all attributes in the PIMA**

**Indian Diabetes Dataset where blue and orange colour distribution**

**respectively denotes non-diabetes and diabetes class.**

Preprocessing, in the proposed pipeline, is the heart of achieving the state-of-the-art result, which consists of outlier rejection, filling missing values, data standardization, feature selection.

Modules to be Implemented are:

* Preprocessing(Outlier Rejection, Filling Missing Values, Feature Selection)
* K-Fold Cross Validation
* ML Models
* k-Nearest Neighbour
* Decision Tree
* AdaBoost
* Random Forest
* Naïve Bayes
* XGBoost
* Ensembling ML Models
* Multi-Layer Perceptron(MLP)

**5.1.1 Preprocessing**

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models.

Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model. Preprocessing of raw data is the integral step in the proposed pipeline, as the quality of data can drive the classifiers to learn directly.

The preprocessing step includes Outlier rejection (P), Filling missing values (Q), Data Standardization(R), Feature selection of the attribute

* + - 1. **Outlier Rejection(P):**

Text

Description automatically generatedThe outlier is a markedly deviated observation from other observations. It requires to be rejected from data distribution as the classifiers are very much sensitive to the data range and distribution of the attributes. The mathematical formulation for the outlier rejection in this literature can be written as

x is the instances of the feature vector that lies in n dimensional space, x ∈ Rn . Q1, Q3, and IQR is the first quartile, third quartile, and interquartile range of the attributes respectively,

where Q1, Q3, IQR ∈ Rn .

* + - 1. **Filling Missing Values(Q):**

The attributes, after outlier rejection, were processed to fill the missing or null values as they could lead to the wrong prediction for any classifiers. In the proposed framework, the missing or null values were imputed by the mean values of the attributes rather than dropping . The imputation with the mean is beneficial as it imputes the continuous data without introducing outliers.

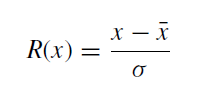
Text

Description automatically generated with medium confidence

where *x* is the instances of the feature vector that lies in *n*-dimensional space, *x * R*n*.

* + - 1. **Data Standardization(R) :**

Data standardization is the process of converting data to a common format to enable users to process and analyze it. The standardization or Z-score normalization is the technique to rescale the attributes for achieving standard normal distribution with zero mean and unit variance. The standardization reduces the skewness of the data distribution.



where *x* is the *n*-dimensional instances of the feature vector, *x * R*n*. *x * R*n* and   R*n* are the mean and standard deviation of the attributes. However, in many ML models such as tree-based models are probably the models, where feature standardization can't provide a guarantee for significant improvement.

The accuracy of the classifiers increases with the increment of the attribute’s dimension. However, the performance of the classifiers will tend to reduce when the attribute’s dimension increases without increasing the samples. Such a scenario, in machine learning, is referred to as a **curse of dimensionality**.

Due to a curse of dimensionality, the space of the feature becomes sparser and sparser which forces the classifiers to be over fitted by loosing generalizing capability. The curse of dimensionality basically means that the error increases with the increase in the number of features. It refers to the fact that algorithms are harder to design in high dimensions and often have a running time exponential in the dimensions.

* + - 1. **Feature Selection**

To compare the performance of the PID dataset three most commonly used methods for feature selection are Principle Component Analysis (PCA), Independent Component Analysis (ICA), Correlation-based technique

**5.1.1.4.1. Principle Component Analysis(PCA)**

Principal Component Analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation. These new transformed features are called the Principal Components.

It is a method that uses simple matrix operations from linear algebra and statistics to calculate a projection of the original data into the same number or fewer dimensions. PCA generally tries to find the lower-dimensional surface to project the high-dimensional data. It is a feature extraction technique, so it contains the important variables and drops the least important variable.

**Steps for PCA algorithm**

**Step 1 : Getting the dataset --** Firstly, take the input dataset and divide it into two subparts X and Y, where X is the training set, and Y is the validation set.

**Step 2 : Representing data into a structure --** Now represent the dataset into a structure. Such as represent the two-dimensional matrix of independent variable X. Here each row corresponds to the data items, and the column corresponds to the Features. The number of columns is the dimensions of the dataset.

**Step 3 : Standardizing the data --** In this step , standardize the dataset. Such as in a particular column, the features with high variance are more important compared to the features with lower variance.

If the importance of features is independent of the variance of the feature, then divide each data item in a column with the standard deviation of the column. Here name the matrix as Z.

**Step 4 : Calculating the Covariance of Z --**  To calculate the covariance of Z, take the matrix Z and transpose it. After transpose, multiply it by Z. The output matrix will be the Covariance matrix of Z.

**Step 5 : Calculating the Eigen Values and Eigen Vectors --** Now need to calculate the eigenvalues and eigenvectors for the resultant covariance matrix Z. Eigenvectors or the covariance matrix are the directions of the axes with high information. And the coefficients of these eigenvectors are defined as the eigenvalues.

**Step 6 : Sorting the Eigen Vectors --** In this step , take all the eigenvalues and will sort them in decreasing order, which means from largest to smallest simultaneously sort the eigenvectors accordingly in matrix P of eigenvalues. The resultant matrix will be named as P\*.

**Step 7 : Calculating the new features Or Principal Components --**Here we will calculate the new features. To do this, we will multiply the P\* matrix to the Z. In the resultant matrix Z\*, each observation is the linear combination of original features. Each column of the Z\* matrix is independent of each other.

**Step 8 : Remove less or unimportant features from the new dataset --** The new feature set has occurred, so we will decide here what to keep and what to remove. It means, we will only keep the relevant or important features in the new dataset, and unimportant features will be removed out.

**PCA Algorithm**

**Input:** The original n-dimensional data, X € Rn with N number of sample and variance threshold,

Tvariance

**Output:** The reduced k-dimensional data, Y € Rk

1 Load X € Rn and compute it's mean, ,

where € Rn

2 Compute the n x n covariance matrix,

Cn\*n = T

3 Compute eigen decomposition of Cn\*n as PDP-1, where P € Rn is the matrix of eigen vectors and Dn\*n is the diagonal matrix with eigenvalues on the diagonal

4 Sort the eigen vectors by descending order to choose first k eigen vectors that will have variance ≥ Tvariance and form a new projection matrix, Wn\*k

5 Project data X into a new k-dimensional space by Y = WTX, where Y € Rk

**5.1.1.4.2 Independent Component Analysis(ICA)**

Independent Component Analysis (ICA) is a machine learning technique to separate independent sources from a mixed signal. Unlike principal component analysis which focuses on maximizing the variance of the data points, the independent component analysis focuses on independence, i.e. independent components. Using ICA, we can extract the desired component from the amalgamation of multiple signals.

**Steps for ICA Algorithm:**

At a high level, ICA can be broken down into the following steps.

**Step 1 :** Center x by subtracting the mean

**Step 2 :** Whiten x

**Step 3 :** Choose a random initial value for the de-mixing matrix w

**Step 4 :** Calculate the new value for w

**Step 5 :** Normalize w

**Step 6 :** Check whether algorithm has converged and if it hasn’t, return to step 4

**Step 7 :** Take the dot product of w and x to get the independent source signals S=W.x

“Whiten”a given signal means that transform it in such a way that potential correlations between its components are removed (covariance equal to 0) and the variance of each component is equal to 1. Another way of looking at it is that the covariance matrix of the whitened signal will be equal to identity matrix.

The actual way we set about whitening a signal involves the eigen-value decomposition of its covariance matrix.  Once finished preprocessing the signal, for each component, we update the values of the de-mixing matrix w until the algorithm has converged or the maximum number of iterations has been reached. Convergence is considered attained when the dot product of w and its transpose is roughly equal to 1

**ICA Algorithm**

**Input:** The original n-dimensional data, X€ Rn

**Output:** The reduced k-dimensional data, Y € Rk

1 Set non-quadratic nonlinear function, G for the approximation of neg-entropy

2 Initialize W of W x H = X, where W, H, and X are the ratios of the sources during mixing, the matrix containing the different components, and the mixed output respectively.

3 Perform PCA on X by X = PCA(X) as in IV-A

4 while W changes do

5 W = mean(X \* G(W.X)) - mean(G'(WT.X)),

where G' is the first derivative of non-quadratic nonlinear function, G

6 W = orthogonalize(W)

1. Compute, Y = W.X, where Y € Rk

**5.1.1.4.3 Correlation - Based Feature Selection**

Correlation is a statistical measure that expresses the extent to which two variables are linearly related (meaning they change together at a constant rate). It’s a common tool for describing simple relationships without making a statement about cause and effect.

Correlation is measured with the help of Correlation Coefficient “r”. Correlation coefficients are used to measure the strength of the linear relationship between two variables. A correlation coefficient greater than zero indicates a positive relationship while a value less than zero signifies a negative relationship. A value of zero indicates no relationship between the two variables being compared.

**Correlation – Based Feature Selection Algorithm**

**Input:**

The original n-dimensional data, X€ Rn and expected outcome, YT € R

**Output:** The reduced k-dimensional data, Y € Rk

1 for i ≤n do

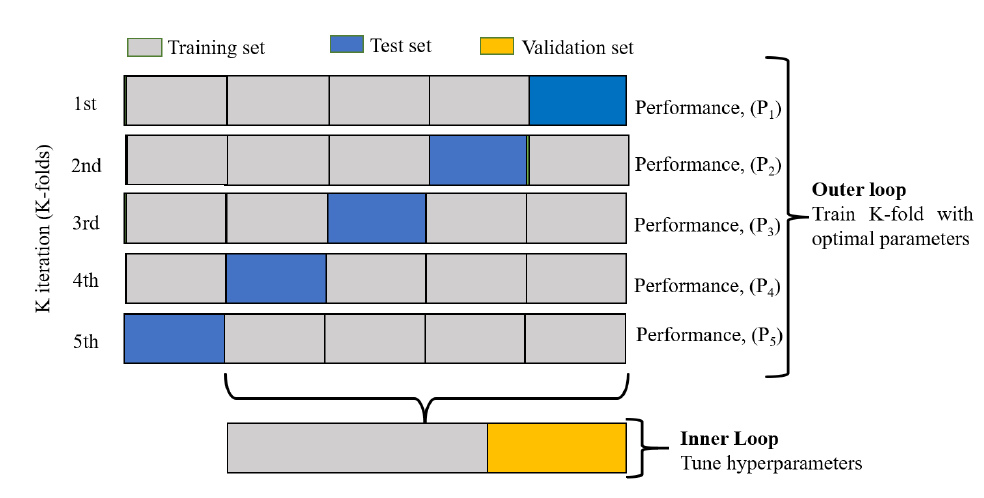
2 riT =

3 Sort the correlation, riT by descending order to

Choose first k features for Y € Rk

**5.1.2. K-Fold Cross Validation**

The K-fold Cross-validation (KCV) technique is one of the most widely used approaches by practitioners for model selection and error estimation of classifiers.

 The pictorial presentation of the data splitting (5-fold cross-validation),used in this literature is

**Figure 5.1.2.1: The partitioning of the PID dataset for KCV for both the**

**hyperparameters tuning and evaluation**.

The PID dataset has partitioned into K folds. The K −1 folds are used to train and fine-tune the hyperparameters in the inner loop where the grid search algorithm was employed. In the outer loop (K times), the best hyperparameters and the test data were used to evaluate the model.

Since the PID dataset contains an imbalanced positive and negative samples, the stratified KCV has been used to preserve the percentage of samples for each class as same as in the original percentage. M is the final performance metric for the classifiers and Pn ∈ R, n = 1, 2, . . . ,K is the performance metric for each fold.

A picture containing text, clock, antenna

Description automatically generated

* + 1. **ML Models**

Different Machine Learning Models are used such as:

k-Nearest Neighbour(k-NN)

Decision Tree(DT)

AdaBoost(AB)

Random Forest(RF)

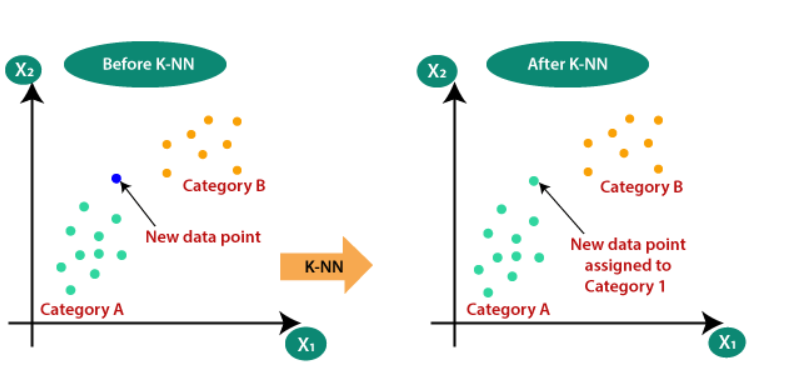
Naïve Bayes(NB)

XGBoost(XB)

**5.1.3.1. k-Nearest Neighbor (k-NN):**

k-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique. k-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.

k-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using k- NN algorithm. It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.



**Figure 5.1.3.1.1 : Working of k-Nearest Neighbors**

The K-NN working can be explained on the basis of the below steps:

**Step-1:** Select the number K of the neighbors

**Step-2:** Calculate the Euclidean distance of K number of neighbors

**Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.

**Step-4:** Among these k neighbors, count the number of the data points in each category.

**Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.

**Step-6:** Model is ready.

**k-NN Algorithm**

**Input:** The n-dimensional data, X€ Rn and target outcome, Y € R

**Output:** The posterior probability, P € [0, 1] of unseen test data, x, where = 1 and C = 2 (diabetes present (C₁) or not (C₂))

1 Calculate geometric distances, Dh for k query points,

Dh = q, where Xi = current instance, xi = query instance, q = order [47].

2 Form a set, S with closest k points

3 Estimates the posterior probability, P for each class

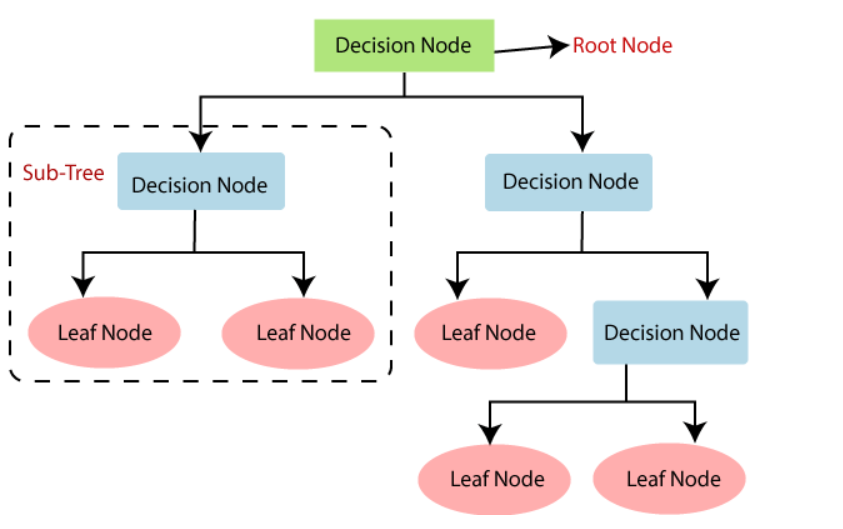
P(C = j | X = x) = 1/k ,

where f(x) is the indicator function to assign the class (1 when patient having diabetes and 0 otherwise)

**5.1.3.2. Decision Tree:**

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given dataset. A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.



**Figure 5.1.3.2.1: Construction of Decision Tree**

**Decision Tree Construction Steps:**

**Step-1:** Begin the tree with the root node, says **S**, which contains the complete dataset.

**Step-2:** Find the best attribute in the dataset using Attribute Selection Measure (ASM).

**Step-3:** Divide the S into subsets that contains possible values for the best attributes.

**Step-4:** Generate the decision tree node, which contains the best attribute.

**Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Attribute Selection Measures:

* Information Gain
* Gini Index

1. **Information Gain:** Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.

Information Gain= Entropy(S)- [(Weighted Avg) \*Entropy(each feature)

**Entropy:** Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data.

**2. Gini Index:** Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.

Gini Index= 1- ∑jPj2

**Decision Tree Algorithm**

**Input:** The n-dimensional data, X € Rn and target outcome, Y€ R

**Output:** The posterior probability, P € [0, 1] of unseen test data, x, where and C = 2 (diabetes present (C₁) or not (C₂))

1 Split ϴ = (j, tm) into Qleft(ϴ) and Qright(ϴ) subsets, where ϴ consisting of a feature, j and threshold, tm

2 Compute the impurity at kth node using an impurity function (H),

G(Q, ϴ) = H (Qleft(ϴ))+ H (Qright (ϴ)), where

H= x (1 – PmC) or

H= - x log(pmC) and

PmC = (yi = C)

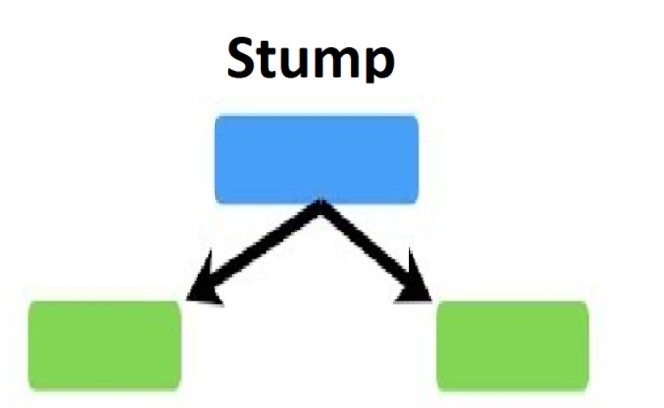
3 Minimise the impurity by selecting the parameters, ϴ\* = argming ϴ G(Q, ϴ)

4 Repeat the above processes for subsets Qleft (ϴ\*) and Qright (ϴ\*) until depth reach to Nm< minsamples or Nm = 1

**5.1.3.3. AdaBoost:**

AdaBoost algorithm, short for Adaptive Boosting, is a Boosting technique that is used as an Ensemble Method in Machine Learning. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights to incorrectly classified instances.

Boosting is used to reduce bias as well as the variance for supervised learning. It works on the principle where learners are grown sequentially. Except for the first, each subsequent learner is grown from previously grown learners. In simple words, weak learners are converted into strong ones.

 The first model is made and the errors from the first model are noted by the algorithm, the record which is incorrectly classified is given as the input for the next model. This process is repeated until the specified condition is met. As you can see in the figure, there are *n* number of models made by taking the errors from the previous model. This is how boosting works. In AdaBoost, the algorithm only makes a node with two leaves, and this is known as Stump.

**Figure 5.1.3.3.1: Stump of AdaBoost**

**Procedure:**

**Step 1 – Creating First Base Learner**: The algorithm takes the first feature, i.e., feature 1, and creates the first stump f1. It will create the same number of stumps as the number of features.  From all these stumps it will create decision trees and can be called stumps base learner model. Out of these all models, the algorithm selects only one. For selecting a base learner, there are two properties, those are, Gini and Entropy. The stump that has the least value will be the first base learner.

**Step 2 – Calculating the Total Error (TE)** : The total error is the sum of all the errors in the classified record for sample weights.

**Step 3 – Calculating Performance of Stump** : Formula for calculating Performance of Stump is: –

Performance of stump = ½ ln[ [1-TE]/TE]

where, ln is natural log and TE is Total Error.

**Step 4 – Updating Weight :** For incorrectly classified records the formula is:

New Sample Weight = Sample Weight \* e^(Performance)

And for correctly classified records, we use the same formula with a negative sign with performance, so that the weight for correctly classified records will reduce compared to the incorrect classified ones. The formula is:

New Sample Weight = Sample Weight \* e^- (Performance)

**Step 5 – Creating New Dataset** : Now, it’s time to create a new dataset from our previous one. In the new dataset, the frequency of incorrectly classified records will be more than the correct ones.

While considering these normalized weights, have to create a new dataset and that dataset is based on normalized weights. It will probably select the wrong records for training purposes. That will be the second decision tree/stump. To make a new dataset based on normalized weight, the algorithm will divide it into buckets.

How does the algorithm decide output for test data?

* Suppose with the dataset, the algorithm constructed 3 decision trees or stumps, the test dataset will pass through all the stumps which have been constructed by the algorithm.
* While passing through the 1st stump, it gives the output as 1, passing through 2nd stump it again gives the output as 1, and while passing through 3rd stump it gives the output as 0.
* So, in AdaBoost algorithm also, the majority of votes take place between the stumps, the same as in random trees. And in this case, the final output will be 1. This is how the output with test data is decided.

**AdaBoost Algorithm**

**Input:** The n-dimensional data, X € Rn with N number of sample and target outcome, Y€ R

**Output:** The posterior probability, P € [0, 1] of unseen test data, x, where and C = 2 (diabetes present (C₁) or not (C₂))

1 Initialize weight sample, D(i) = 1/N, where

i=1,2,..., N

2 for t≤T(n\_Classifiers) do

3 Train a weak learner using distribution Dt, [48].

4 Select a weak hypothesis, ht

Rn → R with low weight error, = Pri ~Dt [ht(xi) ]

5 Choose αt = 1/2\* In() and update,

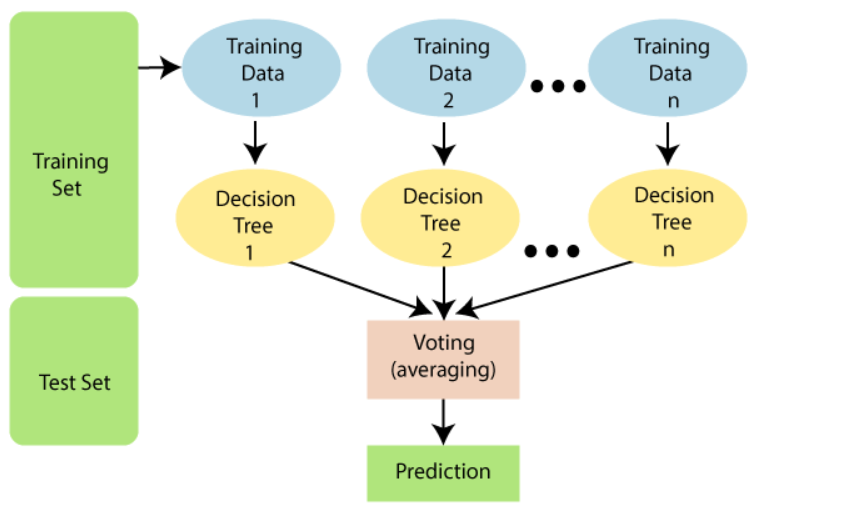
Dt+1(i) = where i = 1, ..., N and zt is the normalization factor.

6 Output posterior probability: P(x) = sign(ht (x))

**5.1.3.4. Random Forest:**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, *"*Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset*."* Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.



**Figure 5.1.3.4.1: Building Random Forest**

The random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

* There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
* The predictions from each tree must have very low correlations.

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase. The Working process can be explained in the below steps:

**Step-1:** Select random K data pointsfrom the training set.

**Step-2:** Build the decision trees associated with the selected data points (Subsets).

**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4:** Repeat Step 1 & 2.

**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

**Random Forest Algorithm**

Input: The n-dimensional data, X € Rn and target outcome,

Y€ R

Output: The posterior probability, P €[ [0, 1] of unseen C test data, x, where and C = 2

(diabetes present (C₁) or not (C₂))

1 for b= 1 to N (n\_Bagging) do

2 Draw a bootstrap sample, (Xb, Yb) from given

(X € Rn, Y€ R)

3 Grow a random-forest tree Tb using Xb and Yb by repeating recursively using the following steps until the minimum node size is nmin.

1) Randomly select m variables from the given n variables.

2) Pick the best variable or split-point among the m variables

3) Split the node into two daughter nodes

Output the ensemble of trees will be {Tb}1N

4 The posterior probability, PRFN(x) = Voting {Pk(x)}1N where P(x) is the class prediction of the kth random-forest.

**5.1.3.5 Naïve Bayes:**

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as: **Naïve**: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. **Bayes**: It is called Bayes because it depends on the principle of Bayes' Theorem.

**Bayes' Theorem:** Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.The formula for Bayes' theorem is given as:

P(Ci |X) =

The fundamental Naïve Bayes assumption is that each feature makes an: Independent and equal contribution to the outcome.

**Naïve Bayes Procedure:** Convert the given dataset into frequency tables. Generate Likelihood table by finding the probabilities of given features. Now, use Bayes theorem to calculate the posterior probability.

**Naïve Bayes Algorithm**

Input: The n-dimensional data, X € Rn and target outcome, Y€ R

Output: The posterior probability, P € [0, 1] of unseen C test data, x, where and C = 2 (diabetes present (C₁) or not (C₂))

1 Compute the prior probabilities for each of the class [49],

P( = C₁) = and P( = C₂) =

where N is the number of sample

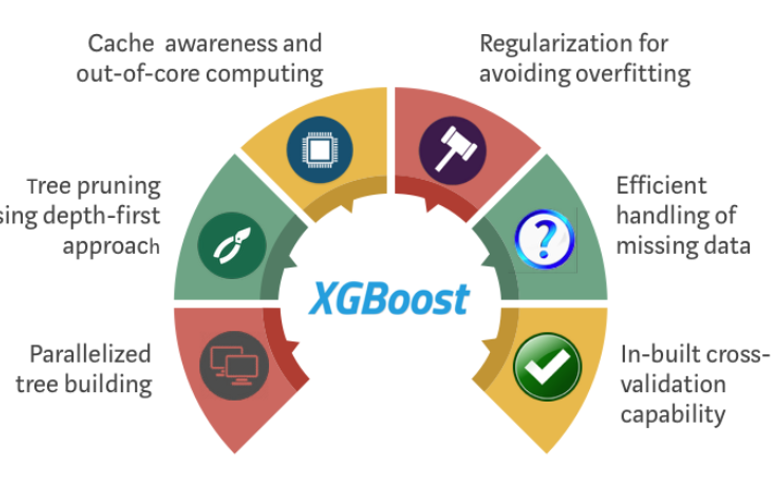
2 The output posterior probability of class for the given predictor (attributes),

P(Ci |X) =

where P(X| Ci) is the likelihood of the predictor for a given class and P(X) is the prior probability of predictor

**5.1.3.6 XGBoost:(Extreme Gradient Boosting)**

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks. However, when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class right now.



**Figure 5.1.3.6.1 :XGBoost Features**

**Procedure:**

**Step 1:** Install libraries, xgboost, margrittr, Matrix

**Step 2:** Create a Matrix for train and test datasets- with the use of xgb.DMatrix() function

**Step 3:** Set parameters, for params and watchlist

**Step 4:** Build model using xgb.train() function

**Step 5:** Use xgb.importance() function for feature analysis

**Step 6:** Make prediction with the use of predict() function

**XGBoost Algorithm**

Input: The n-dimensional data, X€ R" and target outcome, Y€ R

Output: The posterior probability, P € [0, 1] of unseen C test data, x, where and C = 2 (diabetes present (C₁) or not (C₂))

1 Initialize the model with constant value:

F0(x) = argmin (Y,)[32], where L(Y, F(x)) is the differentiable loss function and N is the number of sample

2 for m = 1 to M (n\_Iterations) do

3 Compute pseudo-residuals, rim = -

Where i = 1,2,3…..N

4 Fit a base tree. hm using training set (Xi,rim) for

I = 1,2…N

5 Compute multiplier m by

m = argmin(Yi,Fm-1 (Xi) + hi(Xi))

6 Update the model by Fm(x) = Fm-1 (x)+ mhm(x)

7 Fm(x) is the desired posterior probability, p€[0,1]

**5.1.3.7. HyperParameter Tuning:**

A model hyperparameter is a characteristic of a model that is external to the model and whose value cannot be estimated from data. The value of the hyperparameter has to be set before the learning process begins. For example, c in Support Vector Machines, k in k-Nearest Neighbors, the number of hidden layers in Neural Networks. Grid-search is used to find the optimal hyperparameters of a model which results in the most ‘accurate’ predictions.

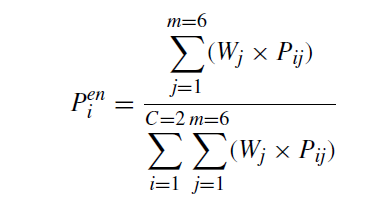
**Table 5.1.3.7.1: Different ML models with hyperparameters to be tuned by the grid search technique in the inner loop**

|  |  |
| --- | --- |
| **ML Models** | **Hyperparameters** |
| K-NN | 1. Number of neighbors for queries 2. Computing algorithm for nearest neighbors  * **Ball Tree(BT):** Node defines a D-Dimensional hypersphere or ball * **KD Tree(KDT):** Leaf node is a D-dimensional point * **Brute:** Based on the brute-force search  1. Leaf size for BT or KDT which depends on the nature of problem 2. Metric (Manhattan distance (L₁-norm) or Euclidean distance (L2-norm)) |
| DT | 1. Measuring function: Gini impurity or Entropy 2. The strategy used to choose the splitst each node 3. The minimum samples for an internal node 4. The minimum samples for a leaf node. |
| RF | 1. The trees in the forest. 2. Measuring function: Gini impurity or Entropy |
| AB | 1. The boosting algorithm (Real boosting or Discrete boosting) 2. Learning rate to shrink the contribution of each classifier 3. The maximum number of estimators to terminate the boosting |
| NB | 1. Portion of the largest variance of the attributes |
| XB | 1. Minimum sum of instance weight (Hessian) 2. Minimum loss reduction for further partitioning on the leaf node 3. Subsample ratio of the training instance 4. Subsample ratio for constructing each tree 5. Maximum tree depth |

* + 1. **Ensembling of ML Classifiers**

Different ML models such as k-NN, DT, AB, RF, NB, and XB have been trained and tested in the proposed framework. The hyperparameters which will tune, in the inner loop. The ensembling of the ML model is the well-known technique to boost the performance using a group of classifiers. In ensembling, the aggregation of the output from different models can improve the precision of the prediction.

The output from each model, *Yj* (*j =* 1,2, 3, …,*m* = 6)  R*C* assigns *C* = 2 (either having diabetes, *C*1 or not, *C*2) confidence values *Pi *  R(*i* = 1, 2) to the unseen test data where *Pi *[0, 1] and . The weighted aggregation of different ML models in this literature was performed using the equation as



where the weight, *Wj* is the corresponding AUC of that *jth* classifier. Since proposing a weighted soft voting ensemble, need an imbalanced, as in the PID dataset, unbiased metric as a weight. That is why choose AUC as a weight for the proposed ensembling classier. The output of the ensembled model, *Y*  R*C* has the confidence values *Peni*  [0, 1]. The final class label of the unseen data, *X * R*n* from ensembled model will be *Ci* if *Peni* = *max*(*Y* (*X*)).

**5.2 Dataset**

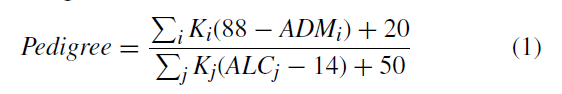
The ML models were trained and tested on publicly available PIMA Indians Diabetes (PID) dataset of 768 female diabetic patients from the Pima Indian population near Phoenix, Arizona.

This dataset consists of 268 diabetic patients (positive) and 500 non-diabetic patients (negative) with eight different attributes. The descriptions of the attributes and brief statistical summary are shown in below Table 5.2.1.

**Table 5.2.1: The overview of Diabetic Patient Cohort**

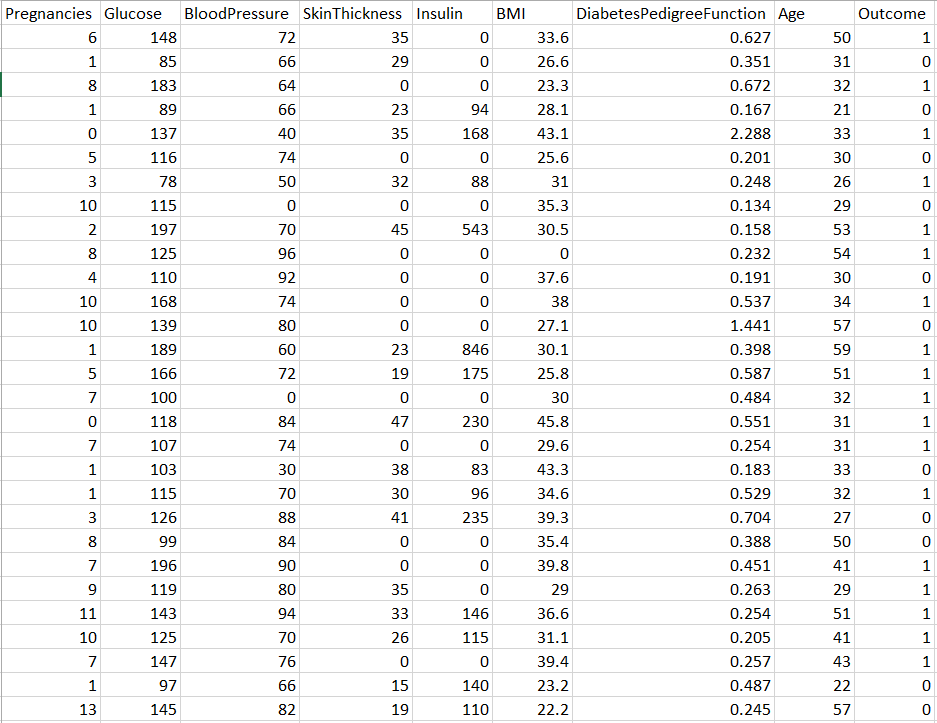
|  |  |  |  |
| --- | --- | --- | --- |
| **S.no** | **Attributes** | **Description** | **Mean ± Std** |
| 1 | Pregnant(F1) | Number of times pregnant | 3.85 **±** 3.37 |
| 2 | Glucose(F2) | Plasma glucose concentration at 2hrs in an Oral Glucose Tolerance test | 120.90**±**31.97 |
| 3 | Pressure(F3) | Diastolic blood pressure(mm Hg) | 69.11±19.36 |
| 4 | Triceps(F4) | Triceps skinfold thickness(mm) | 20.54±15.95 |
| 5 | Insulin(F5) | 2-hour serum Insulin(µU/ml) | 79.81±115.24 |
| 6 | BMI(F6) | Body Mass Index(Weight in kg/(Height in inches)² | 32.00±7.88 |
| 7 | Pedigree(F7) | Diabetes Pedigree Function | 0.47±0.33 |
| 8 | Age (F8) | Age in years | 33.24±11.76 |

The Pedigree (Diabetes Pedigree Function)(F7) was calculated as in (1).



where ‘ i ’ and ‘ j ‘ respectively denote the relatives who had developed and NOT developed diabetes. K is the percentage of shared genes by the relatives (K = 0.500 for the parent or full sibling, K = 0.250 for a half-sibling, grandparent, aunt or uncle and K = 0.125 for a half aunt, half-uncle or first cousin). ‘ ADMi ‘ and ‘ ACLj ‘ is the age of relatives, in years, at the time of diagnosing and at the last non-diabetic test respectively.

Attribute Pregnant (F1) gives no. of times pregnant. Attribute Glucose (F2) says about Plasma Glucose Concentration at 2 Hours in a Oral Glucose Tolerance Test. Pressure (F3) is Diastolic Blood Pressure measured in mm Hg. Triceps (F4) is value of Triceps Skin Fold Thickness in mm. Insulin (F5) gives value of 2-Hour Serum Insulin in mU/ml. BMI(F6) is Body mass Index of a person calculated as Weight in kg/(Height in inches)2. Last Attribute is Age (F8) i.e., Age in years.

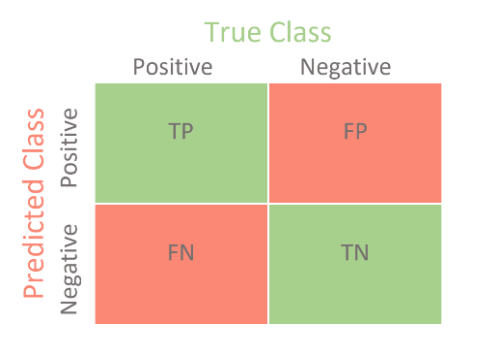
****

**Figure 5.2.2: PIMA Indian Diabetes**

**5.3 Evaluation Metrics**

The models were implemented using the Python programming language with different Python and Keras APIs. The experiments were carried out on a machine running *Windows-10* operating system with the following hardware configuration:Intel ® CoreTM i7-7700 HQ CPU@2.80 *GHz* processor with Install memory (RAM): 16.0*GB* and GeForce GTX 1060 GPU with 6*GB* GDDR5 memory. All the extensive experiments were evaluated using several metrics where each metric has a different meaning of evaluation.

The **confusion matrix** of True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN) along with different metrics.



**Figure 5.3.1 : Confusion Matrix**

The **Sensitivity(Sn)** used to quantify the type-II error(the patient having positive symptoms, but erroneously fails to be rejected)

Sn = TP/(TP + FN)

The **Specificity(Sp)** used to quantify type-I error (the patient having negative symptoms, but detected as positive).

Sp = TN/(TN + FP)

**Precision (Pr)**, have been used to evaluate the percentage of correctly classified diabetes patients having positive conditions.

Pr **=**  TP/(TP + FP)

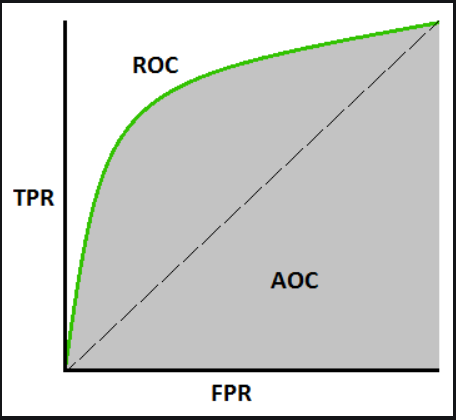
**False Omission Rate (FOR)** is the proportion of the individuals with a negative test result, for which the true condition is positive.

FOR = FN/(TN+FN)

**Diagnostic Odds Ratio (DOR)** the effectiveness of a diagnostic test.

DOR = (TP/FN)/(FP/TN)

The **Receiver Operating Characteristics (ROC)** with Area Under the **ROC Curve (AUC)** is also reported to measure how well predictions are ranked, rather than their absolute values.



**Figure 5.3.2. Area Under Curve**

**6. TESTING**

**6.1 Introduction to Testing**

Testing is a fault detection technique that tries to create failure and erroneous states in a planned way. This allows the developer to detect failures in the system before it is released to the customer.

Note that this definition of testing implies that a successful test is test that identifies faults. We will use this definition throughout the definition phase. Another often used definition of testing is that it demonstrates that faults are not present. Testing can be done in two ways: Top down approach ,Bottom up approach

* Top down Approach: This type of testing starts from upper level modules. Since the detailed activities usually performed in the lower level routines are not provided stubs are written.
* Bottom up Approach: Testing can be performed starting from smallest and lowest level modules and proceeding one at a time. For each module in bottom up testing a short program executes the module and provides the needed data so that the module is asked to perform the way it will when embedded within the larger system. In this project, bottom up approach is used where the lower level modules are tested first and the next ones having much data in them.

Testing Methodologies: The following are the Testing Methodologies:

* Unit Testing.
* Integration Testing.
* User Acceptance Testing.
* Output Testing.

**6.1.1 Unit Testing:** Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a modules control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing. During this testing, each module is tested individually and the module inter- faces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

**6.1.2 Integration Testing:** Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design. The following are the types of Integration Testing:

* Top Down Integration: This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner. In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.
* Bottom-up Integration: This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:
* The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
* A driver (i.e.) the control program for testing is written to coordinate test case input and output.
* The cluster is tested.Drivers are removed and clusters are combined moving upward in the program structure. The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

**6.1.3 User Acceptance Testing:** User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

**6.1.4 Output Testing:** After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways one is on screen and another in printed format.

**6.2 Test Cases**

**Table 6.2.1 Test cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case**  **Name** | **Description** | **Step** | **Expected** | **Actual** | **Test**  **Status (P/F)** |
| Input  Data: 1st Attribute | No. of times pregnant in Integer Value | Predict person is diabetic or not | has to give  the result | result | P |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case**  **Name** | **Description** | **Step** | **Expected** | **Actual** | **Test**  **Status (P/F)** |
| Input  Data: 2nd Attribute | Glucose Concentra-tion in decimal value | Predict person is diabetic or not | has to give  the result | result | P |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case**  **Name** | **Description** | **Step** | **Expected** | **Actual** | **Test**  **Status (P/F)** |
| Input  Data: 3rd Attribute | Diastolic Blood Pressure in mm Hg | Predict person is diabetic or not | has to give  the result | result | P |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case**  **Name** | **Description** | **Step** | **Expected** | **Actual** | **Test**  **Status (P/F)** |
| Input  Data: 4th Attribute | Triceps Skin Fold thickness in mm | Predict person is diabetic or not | has to give  the result | result | P |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case**  **Name** | **Description** | **Step** | **Expected** | **Actual** | **Test**  **Status (P/F)** |
| Input  Data: 5th Attribute | 2 hour Serum Insulin in µU/ml | Predict person is diabetic or not | has to give  the result | result | P |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case**  **Name** | **Description** | **Step** | **Expected** | **Actual** | **Test**  **Status (P/F)** |
| Input  Data: 6th Attribute | Body Mass Index (BMI) in kg/(inch)2 | Predict person is diabetic or not | has to give  the result | result | P |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case**  **Name** | **Description** | **Step** | **Expected** | **Actual** | **Test**  **Status (P/F)** |
| Input  Data: 7th Attribute | Diabetes Pedigree Function in decimal values | Predict person is diabetic or not | has to give  the result | result | P |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case**  **Name** | **Description** | **Step** | **Expected** | **Actual** | **Test**  **Status (P/F)** |
| Input  Data: 8th Attribute | Age in years | Predict person is diabetic or not | has to give  the result | result | P |

**7. RESULTS**

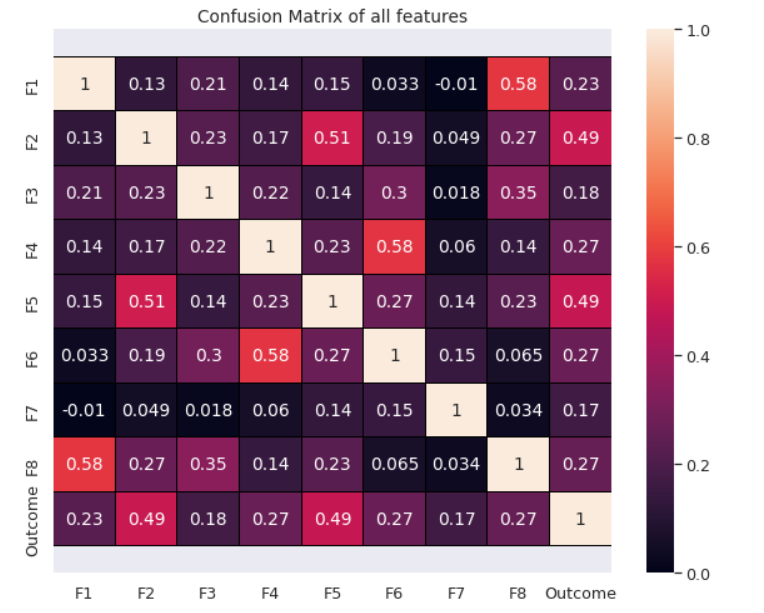
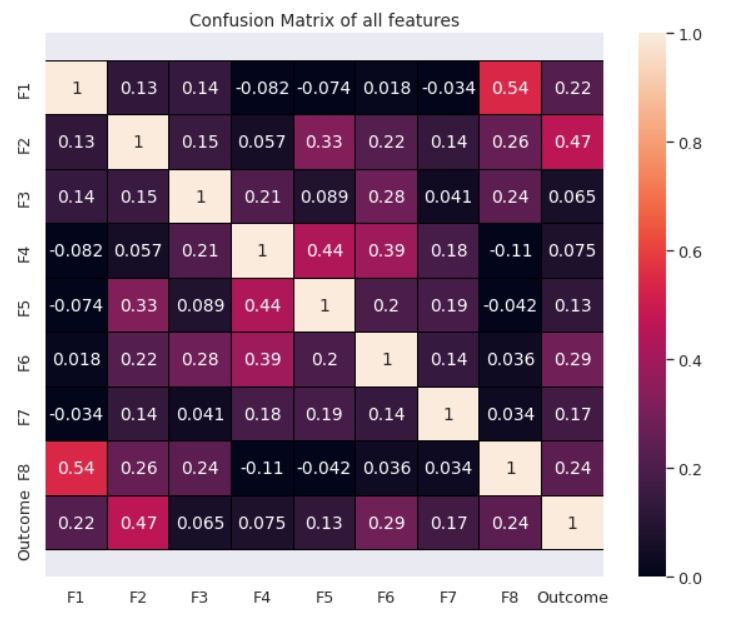
* 1. **Chart, histogram, box and whisker chart

     Description automatically generatedResult of Preprocessing:**

**Chart, box and whisker chart

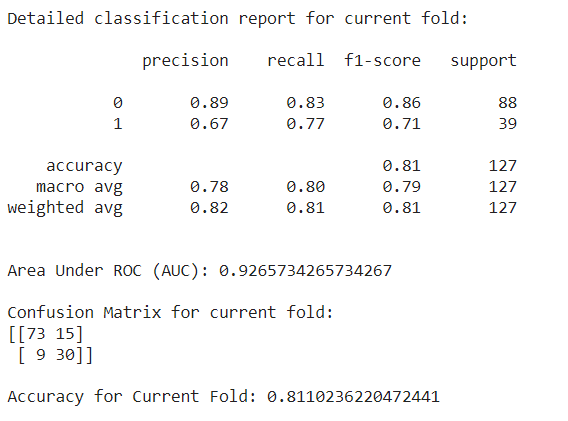
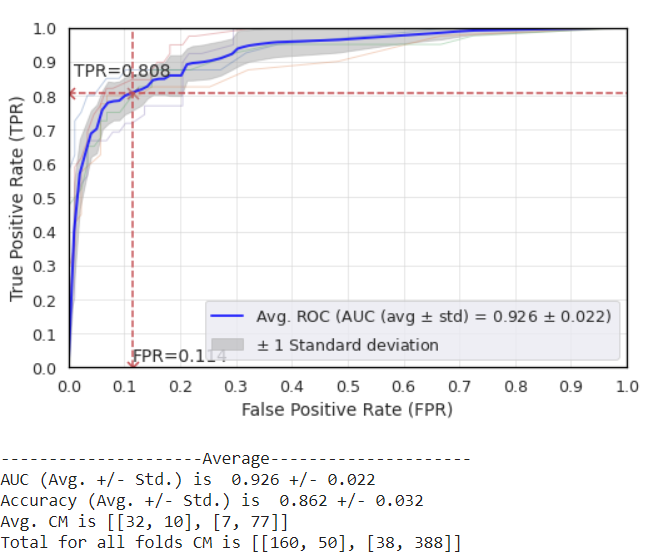
Description automatically generated**

**Figure 7.1.1: Box plot of Attributes**

****

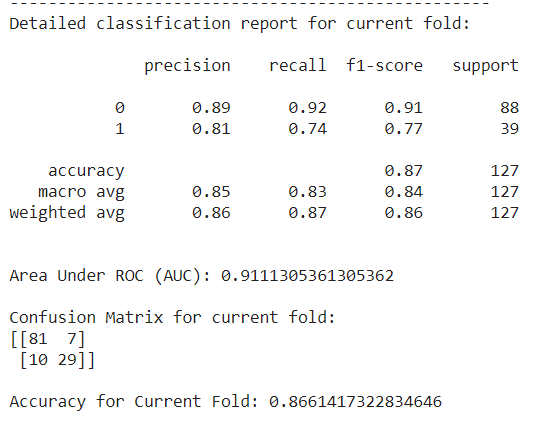
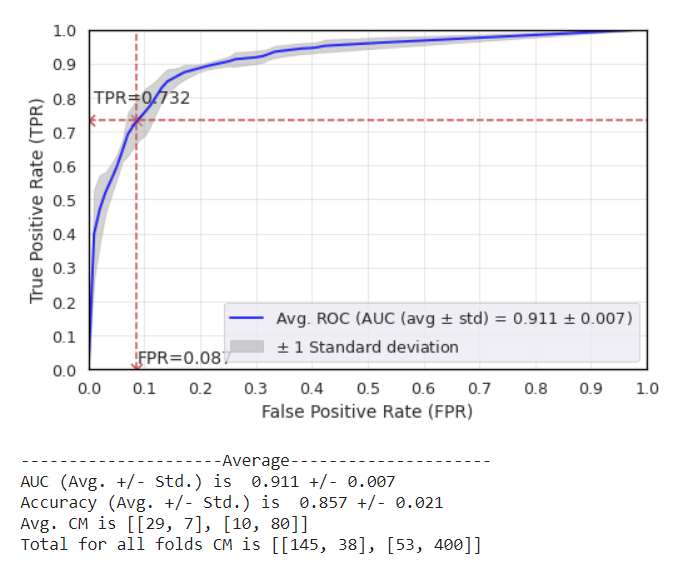
**Figure 7.1.2: Confusion Matrix of Raw Data and after Preprocessing**

* 1. **k-Nearest Neighbor Result:**

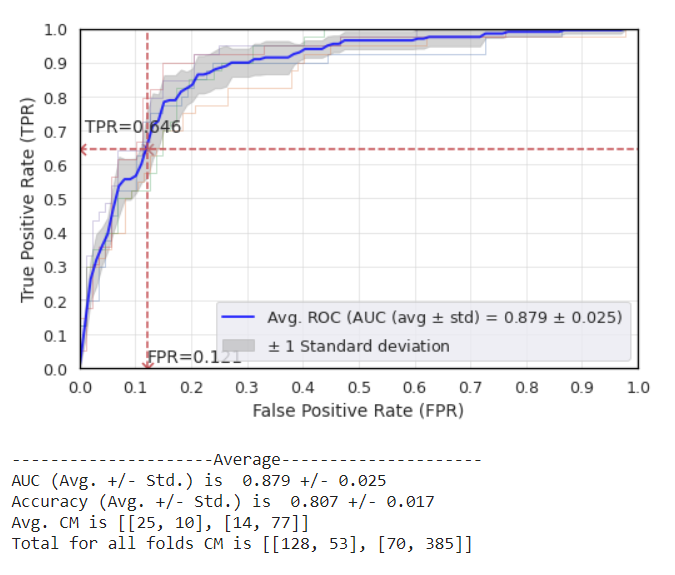
****

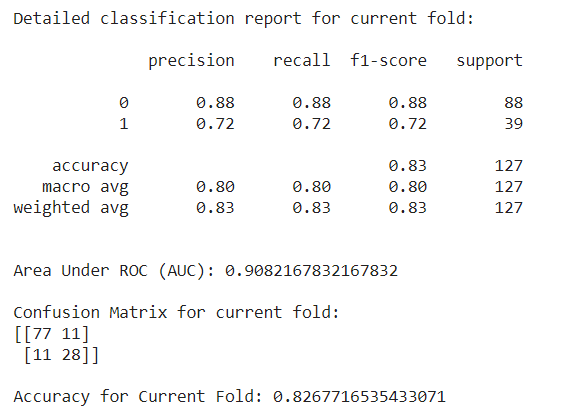
**Figure 7.2.1: Result of kNN**

* 1. **Result of Decision Tree:**

****

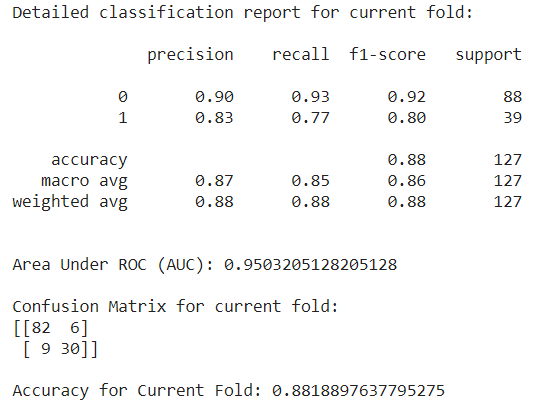
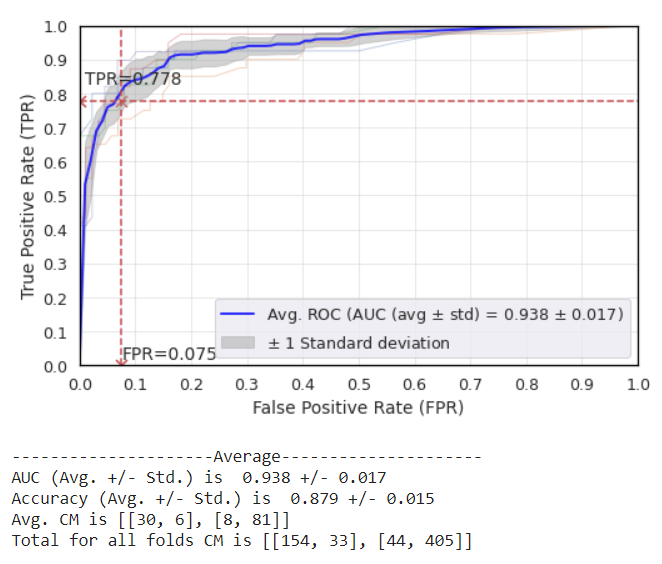
**Figure 7.3.1 Result of Decision Tree**

* 1. **Result of Naïve Bayes:**

****

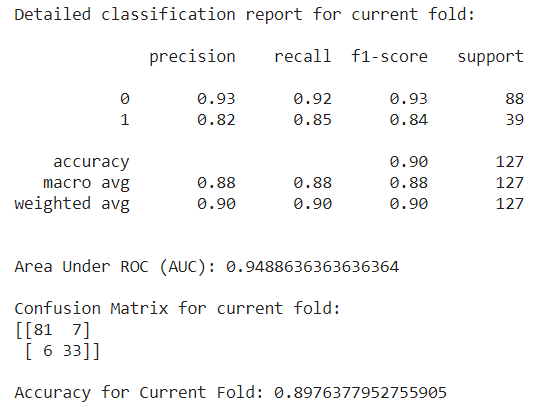
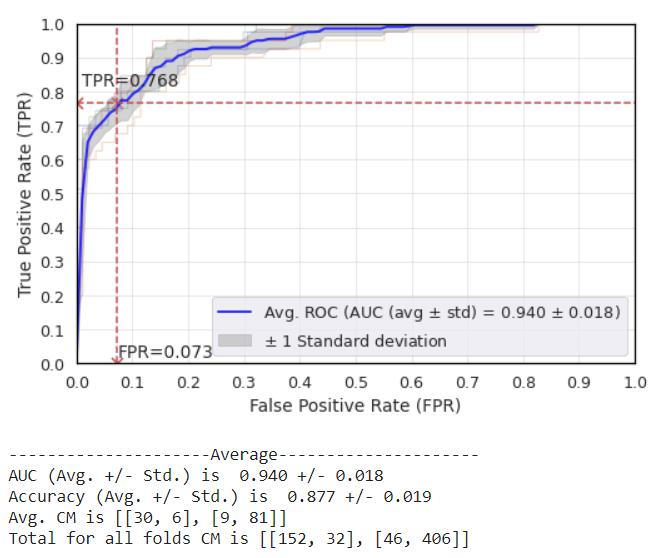
**Figure 7.4.1 Result of Naïve Bayes**

* 1. **Result of Random Forest:**

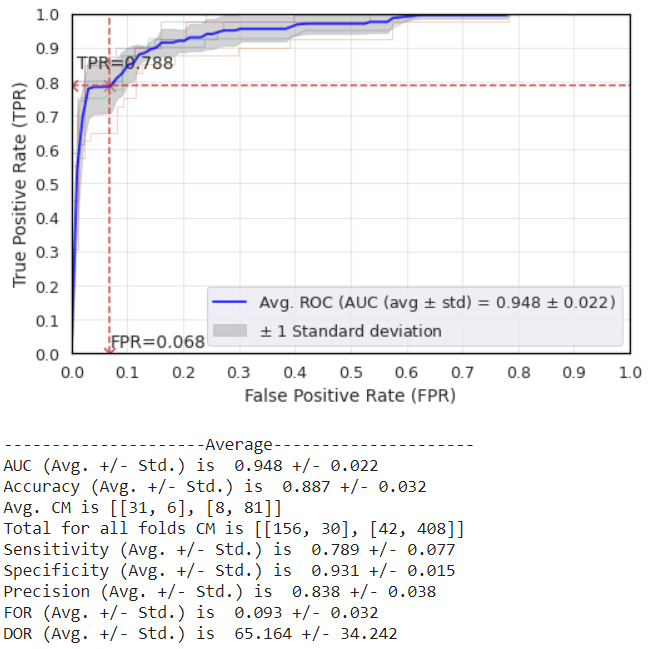
****

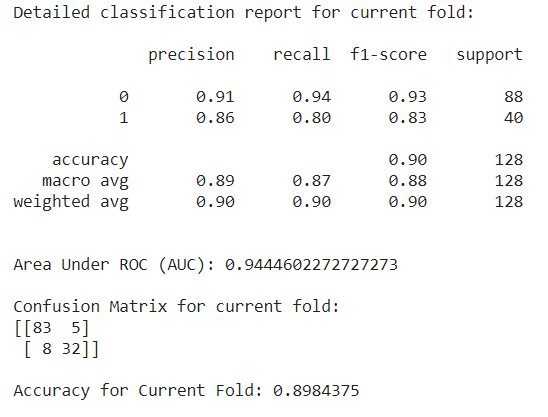
**Figure 7.5.1 Result of Random Forest**

* 1. **Result of AdaBoost:**

****

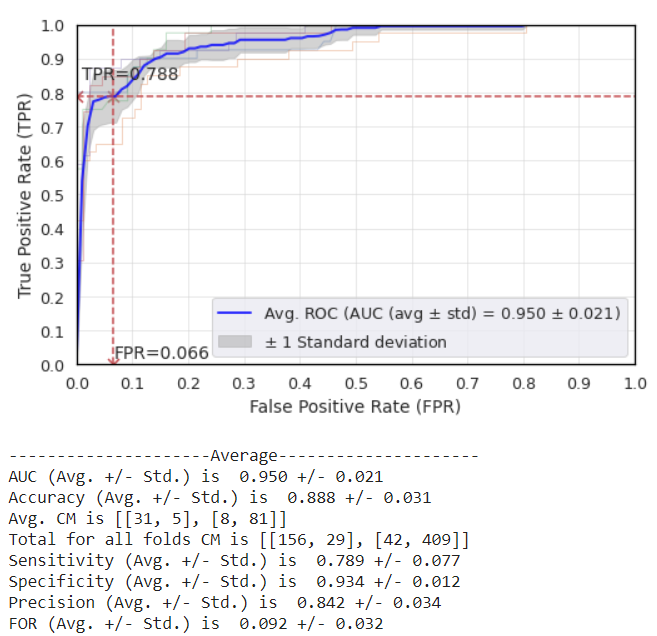
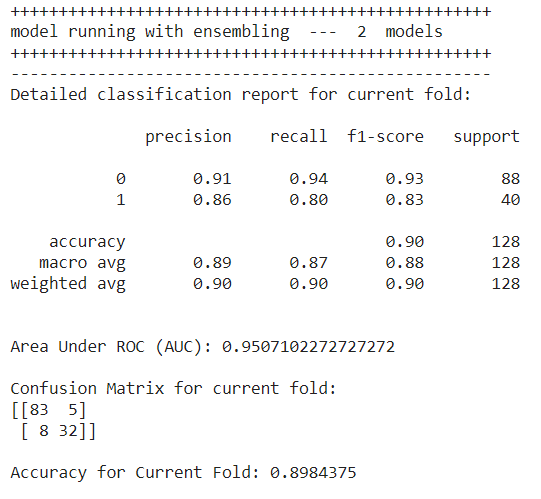
**Figure 7.6.1 Result of AdaBoost**

* 1. **Result of XGBoost:**

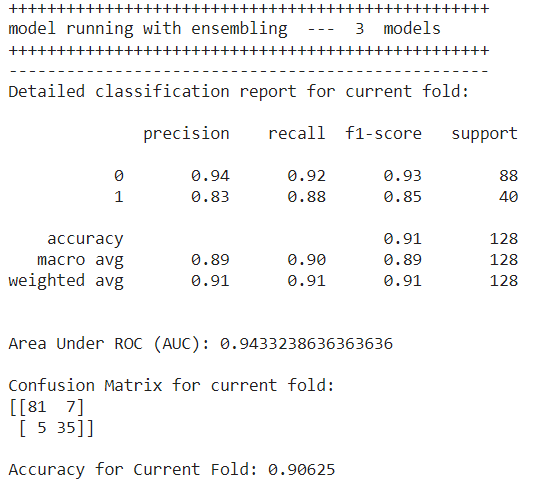
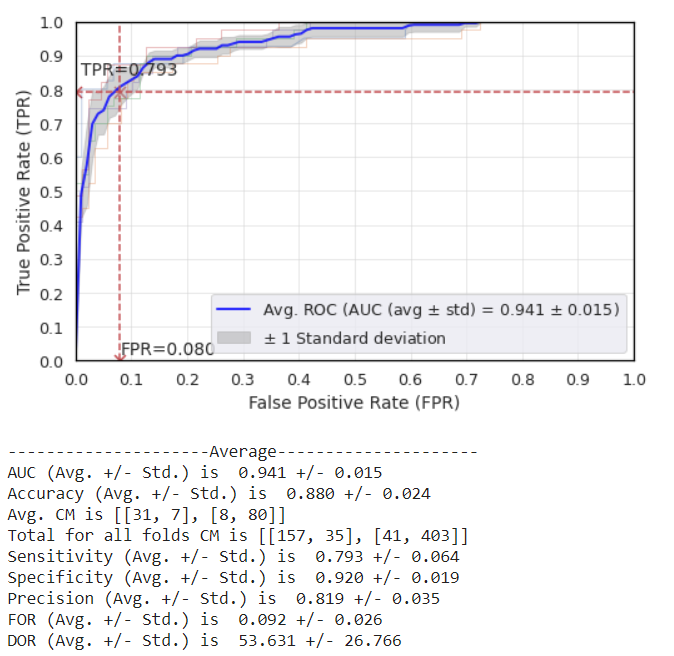
****

**Figure 7.7.1 Result of XGBoost**

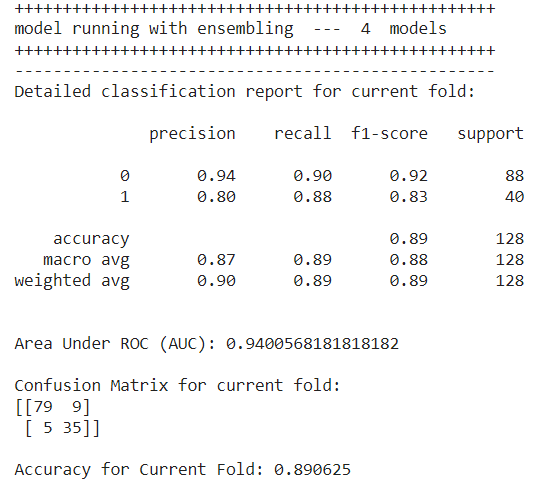
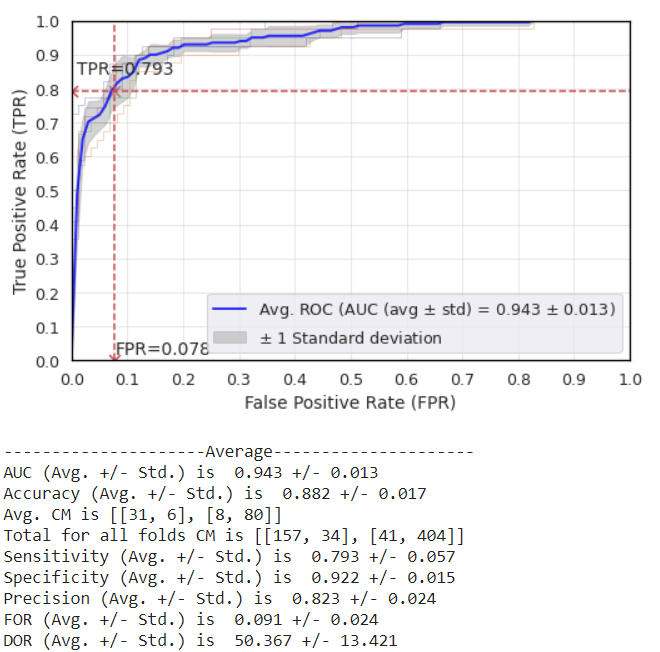
* 1. **Results of Ensembling:**
     1. **Ensembling 2 Models(AB+XB):**

****

**Figure 7.8.1 Result of Ensemblin 2 Models**

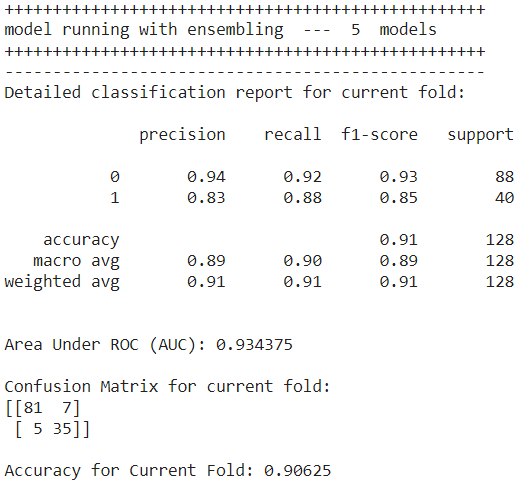
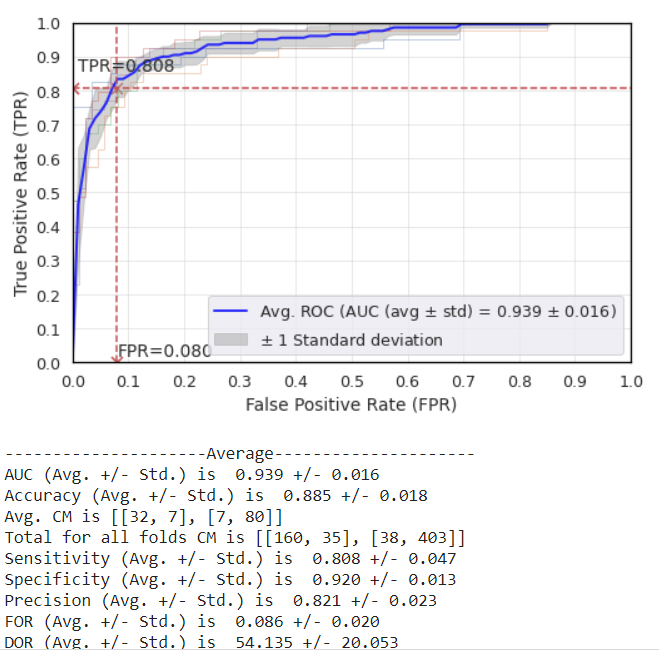
* + 1. **Ensembling 3 Models(kNN+DT+XB):**

**Figure 7.8.2 Result of Ensembling 3 Models**

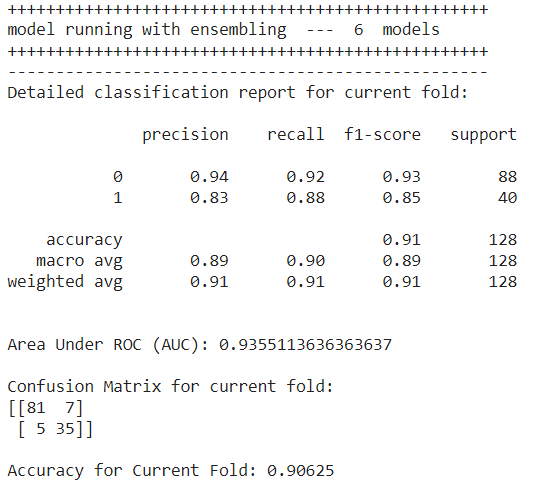
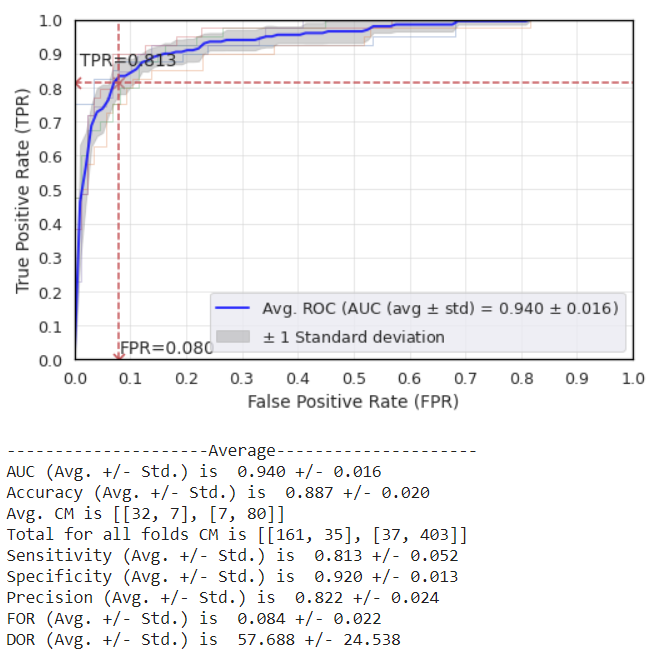
* + 1. **Ensembling 4 Models (DT+AB+RF+XB):**

**Figure 7.8.3 Result of Ensembling 4 Models**

* + 1. **Ensembling 5 Models(kNN+DT+RF+XB+NB):**

****

**Figure 7.8.4 Result of Ensembling 5 Models**

* + 1. **Ensembling 6 Models(kNN+DT+RF+AB+NB+XB)**

**Figure 7.8.5 Result of Ensembling 6 Models**

* 1. **Result Comparision**

Results demonstrates that the proposed weighted-ensemble of AB and XB produces the best prediction for the three metrics out of the five metrics, whereas performs as a second highest with respect to Sp and prevalence independent measurement (DOR). The proposed ensemble model (AB & XB) yields the best performance concerning Sn, FOR, and AUC by improving the XB by the margin of 2.1 %, 0.8 %, and 0.6% respectively. The ensembling model (AB& XB) improves the true-positive rate compare to the XB model alone.

The less FOR values in the ensembling model demonstrates that negative predictive value is high with less Type II error in the diabetes prediction. Furthermore, it is also observed that the proposed ensembling model(AB & XB) yields the best performances for balanced accuracy (average of Sn and Sp) by improving the XB and MLP results by 0.6% and 3.3% respectively, when the proposed preprocessing (*P*+*Q* and correlation-based feature selection) is employed.

The proposed ensembling classier (AB & XB) appears better suited for diabetes prediction from the PID dataset. For ensembling, the base classifiers should have a minimum correlation between them to achieve higher precision in diabetes prediction. The ensembling of two boosting (adaptive (AB) and gradient (XB)) type classifier is the best combination for diabetes prediction. The best combination (AB&XB), along with our proposed preprocessing (*P*+*Q* and correlation-based feature selection), can achieve tremendous success for diabetes prediction in the PID dataset.

**Table 7.9.1 Result Comparision**

|  |  |
| --- | --- |
| **Algorithm** | **Avg ROC(AUC)** |
| k-Nearest Neighbor | 0.926 ± 0.222 |
| Decision Tree | 0.911 ± 0.007 |
| Naïve Bayes | 0.897 ± 0.025 |
| Random Forest | 0.938 ± 0.017 |
| AdaBoost | 0.940 ± 0.018 |
| XGBoost | 0.948 ± 0.222 |
| Ensembling 2 Models(AB+XB) | **0.950 ± 0.021** |
| Ensembling 3 Models(kNN+ DT+XB) | 0.941 ± 0.015 |
| Ensembling 4 Models(DT+AB+RF+XB) | 0.943 ± 0.013 |
| Ensembling 5 Models(kNN +DT+RF+XB+NB) | 0.939 ± 0.016 |
| Ensembling 6 Models(kNN +DT+RF+AB+NB+XB) | 0.940 ± 0.016 |

**8. CONCLUSION AND FUTURE WORK**

**Conclusion**

In this literature, diabetes prediction has been accomplished using the proposed ensemble model from the PID dataset, where the preprocessing plays a crucial role in robust and precise prediction. The quality of the dataset was improved by the proposed preprocessing scheme, where outlier rejection and filling missing values was a core concern.

Such a preprocessing can improve the kurtosis and skewness of the attribute distribution in the PID dataset. The correlation-based attribute selection can improve the correlation between attribute and target outcome whereas PCA and ICA care only the inter-attribute redundancy. In case of tree-based classifier, data standardization can not provide any guarantee to improve the performance.

The robustness validation of the XB, MLP, and proposed ensemble classifier was verified by using the 5-fold cross-validation. Hyperparameters of different classifiers can drive the learning capability of those classifiers, which were optimized using a grid search technique in proposed framework. The AUC as a weight to build a generic ensembling classier is better, as it considers more priority to the model having more AUC.

The comparative results demonstrate that this proposed framework has outperformed other frameworks on AUC, which has shown great potentiality for diabetes prediction from the PID dataset. The ensembling of two boosting type classifiers (AB and XB) is the best combination for diabetes prediction, as the base classifiers should have a minimum correlation between them.

The higher precision in diabetes prediction from the PID dataset using the best combination (AB &XB) can be achieved when proposed preprocessing (*P* + *Q* and correlation-based feature selection) is applied. This ensembling classifier is the best performing classifier with the sensitivity, specificity, false omission rate, diagnostic odds ratio, and AUC as **0.789, 0.934, 0.092, 66.234, and 0.950** respectively which outperforms the state-of-the-art results by **2.00 %** in AUC.

**Future Work**

The proposed trained model will be used to build a web app with a user-friendly interface. Additionally, the proposed framework will be applied to other medical contexts to verify their generality and versatility to predict the disease classes. To tune parameters of classifiers to increase accuracy.

**9. REFERENCES**

**Bibliography**

[1] A. Hyvärinen and E. Oja, ``Independent component analysis: Algorithms and applications,'' *Neural Netw.*, vol. 13, nos. 4\_5, pp. 411\_430, Jun. 2000.

[2] A. K. Dewangan and P. Agrawal, ``Classification of diabetes mellitus using machine learning techniques,'' *Int. J. Eng. Appl. Sci.*, vol. 2, no. 5, pp. 145\_148, May 2015.

[3] A. Misra, H. Gopalan, R. Jayawardena, A. P. Hills, M. Soares, A. A. Reza-Albarrán, and K. L. Ramaiya, ``Diabetes in developing countries,'' *J. Diabetes*, vol. 11, no. 7, pp. 522\_539, Mar. 2019.

[4] A. S. Glas, J. G. Lijmer, M. H. Prins, G. J. Bonsel, and P. M. M. Bossuyt, ``The diagnostic odds ratio:Asingle indicator of test performance,'' *J. Clin.* *Epidemiology*, vol. 56, no. 11, pp. 1129\_1135, Nov. 2003.

[5] B. P. Tabaei and W. H. Herman, ``A multivariate logistic regression equation to screen for diabetes : Development and validation,'' *Diabetes Care*, vol. 25, no. 11, pp. 1999\_2003, Nov. 2002.

[6] D. Cousineau and S. Chartier, ``Outliers detection and treatment: A review,'' *Int. J. Psychol. Res.*, vol. 3, no. 1, pp. 58\_67, Mar. 2010.

[7] D. Sisodia and D. S. Sisodia, ``Prediction of diabetes using classification algorithms,'' *Procedia Comput. Sci.*, vol. 132, pp. 1578\_1585, Jan. 2018.

[8] H. Kaur and V. Kumari, ``Predictive modelling and analytics for diabetes using a machine learning approach,'' *Appl. Comput. Informat.*, Dec. 2018.

[9] I. Jenhani, N. B. Amor, and Z. Elouedi, ``Decision trees as possibilistic classifiers,'' *Int. J. Approx. Reasoning*, vol. 48, no. 3, pp. 784\_807, Aug. 2008.

[10] J. W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler, and R. S. Johannes, ``Using the ADAP learning algorithm to forecast the onset of diabetes mellitus,'' in *Proc. Annu. Symp. Comput. Appl. Med. Care*,Nov. 1988, pp. 261\_265.

[11] The Emerging Risk Factors Collaboration, ``Diabetes mellitus, fasting blood glucose concentration, and risk of vascular disease: A collaborative meta-analysis of 102 prospective studies,'' *Lancet*, vol. 375,pp. 2215-2222, Jun. 2010.

[12] L. Breiman, ``Random forests,'' *Mach. Learn.*, vol. 45, no. 1, pp. 5-32, Oct. 2001.

[13] L. Li, ``Diagnosis of diabetes using a weight-adjusted voting approach,'' in *Proc. IEEE Int. Conf. Bioinf. Bioeng.*, Nov. 2014, pp. 320\_324.

[14] M. Maniruzzaman, M. J. Rahman, M. Al-MehediHasan, H. S. Suri, M. M. Abedin, A. El-Baz, and J. S. Suri, ``Accurate diabetes risk stratification using machine learning: Role of missing value and outliers,'' *J. Med.* *Syst.*, vol. 42, no. 5, p. 92, May 2018.

[15] M. Maniruzzaman, N. Kumar, M. M. Abedin, M. S. Islam, H. S. Suri, A. S. El-Baz, and J. S. Suri, ``Comparative approaches for classification of diabetes mellitus data: Machine learning paradigm,'' *Comput. Methods* *Programs Biomed.*, vol. 152, pp. 23\_34, Dec. 2017.

[16] M. Pradhan and G. R. Bamnote, ``Design of classifier for detection of diabetes mellitus using genetic programming,'' in *Proc. 3rd Int. Conf.* *Frontiers Intell. Comput., Theory Appl.*, Nov. 2015, pp. 763\_770.

[17] N. H. Cho, J. E. Shaw, S. Karuranga, Y. Huang, J. D. da Rocha Fernandes, A. W. Ohlrogge, and B. Malanda, ``IDF diabetes atlas: Global estimates of diabetes prevalence for 2017 and projections for 2045,'' *Diabetes Res.* *Clin. Pract.*, vol. 138, pp. 271\_281, Apr. 2018.

[18] N. Nai-arun and R. Moungmai, ``Comparison of classifiers for the risk

of diabetes prediction,'' *Procedia Comput. Sci.*, vol. 69, pp. 132\_142, Dec. 2015.

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, ``Dropout:Asimpleway to prevent neural networks from overfitting,'' *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929\_1958, Jan. 2014.

[20] P. Cunningham and S. J. Delany, ``k-Nearest neighbour classifiers,'' *Multiple Classifier Syst.*, vol. 34, pp. 1\_17, Mar. 2007.

[21] P. Ramachandran, B. Zoph, and Q. V. Le, ``Searching for activation functions,'' 2017, *arXiv:1710.05941*. [Online]. Available: http://arxiv.org/abs/1710.05941

[22] P. Saeedi, I. Petersohn, P. Salpea, B. Malanda, S. Karuranga, N. Unwin, S. Colagiuri, L. Guariguata, A. A. Motala, K. Ogurtsova, J. E. Shaw, D. Bright, and R. Williams, ``Global and regional diabetes prevalence estimates for 2019 and projections for 2030 and 2045: Results from the international diabetes federation diabetes atlas, 9th edition,'' *Diabetes Res.* *Clin. Pract.*, vol. 157, Nov. 2019, Art. no. 107843.

[23] Q. Wang, W. Cao, J. Guo, J. Ren, Y. Cheng, and D. N. Davis, ``DMP\_MI: An effective diabetes mellitus classi\_cation algorithm on imbalanced data with missing values,'' *IEEE Access*, vol. 7, pp. 102232\_102238, Jul. 2019.

[24] R. Bansal, N. Gaur, and S. N. Singh, ``Outlier detection: Applications and techniques in data mining,'' in *Proc. 6th Int. Conf. Cloud Syst. Big Data*

*Eng. (Conference)*, Jan. 2016, pp. 373\_377.

[25] R. Vaishali, R. Sasikala, S. Ramasubbareddy, S. Remya, and S. Nalluri, ``Genetic algorithm based feature selection and MOE fuzzy classification algorithm on pima indians diabetes dataset,'' in *Proc. Int. Conf. Comput.* *Netw. Informat. (ICCNI)*, Oct. 2017, pp. 1\_5.

[26] S. Arlot and A. Celisse, ``Asurvey of cross-validation procedures for model selection,'' *Statist. Surv.*, vol. 4, pp. 40\_79, Jul. 2010.

[27] S. Bashir, U. Qamar, and F. H. Khan, ``IntelliHealth: A medical decision support application using a novel weighted multi-layer classi\_er ensemble framework,'' *J. Biomed. Informat.*, vol. 59, pp. 185\_200, Feb. 2016.

[28] S. P. Chatrati, G. Hossain, A. Goyal, A. Bhan, S. Bhattacharya, D. Gaurav, and S. M. Tiwari, ``Smart home health monitoring system for predicting type 2 diabetes and hypertension,'' *J. King Saud Univ. Comput. Inf. Sci.*, Jan. 2020.

[29] S. Perveen, M. Shahbaz, A. Guergachi, and K. Keshavjee, ``Performance analysis of data mining classi\_cation techniques to predict diabetes,'' *Procedia Comput. Sci.*, vol. 82, pp. 115\_121, 2016.

[30] T. Chen and C. Guestrin, ``XGBoost: A scalable tree boosting system,'' in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining KDD*, 2016, pp. 785\_79