Designing an Expert System for Diabetic Prediction using Cloud Computing

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering/Information Technology

By

Aryan Chugh (191409)

Under the supervision of

Dr. Rajni Mohana

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

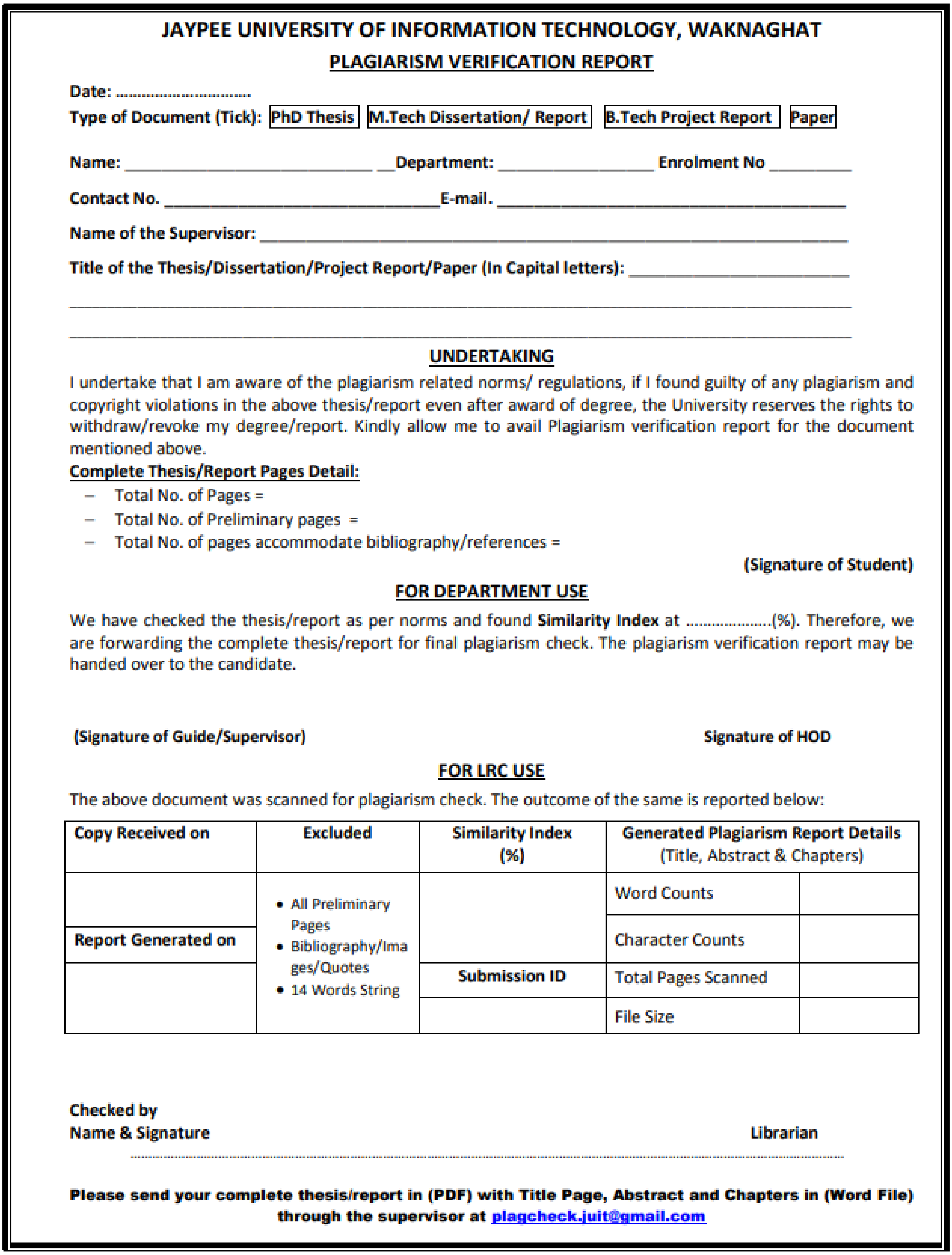
I hereby declare that the work presented in this report entitled **" Designing an Expert System for Diabetic Prediction using cloud computing"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from July 2022 to May 2023 under the supervision of **Dr. Rajni Mohana,** Associate Professor, Department of Computer Science and Engineering and Information Technology.

Aryan Chugh,
191409

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Rajni Mohana
Associate Professor
Department of Computer Science and Engineering and Information Technology
Dated:

Plagiarism Certificate



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: ....

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report | Paper

Name: ______________________ __Department: ______________ Enrolment No ________

Contact No. ______________________________E-mail. ______________________________

Name of the Supervisor: ____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): ____________________

____

____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at ....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor)   Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature   Librarian

....

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

ACKNOWLEDGEMENT

First, I express my heartiest thanks and gratefulness to Almighty God for His divine blessing to make it possible to complete the project work successfully.

I am really grateful and wish my profound indebtedness to Dr Rajni Mohana, Associate Professor, Department of CSE & IT, Jaypee University of Information Technology, Waknaghat. Deep Knowledge & keen interest of my supervisor in the field of “Data Science” to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, and reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

I would like to express my sincere appreciation to Dr Aman Sharma, Assistant Professor in the Department of CSE and IT at the Jaypee University of Information Technology, for their insightful recommendations during the course of my projects.

I would also generously welcome each one of those individuals who have helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents and brother.

Aryan Chugh
Project Group No. 92
Roll No: 191409

TABLE OF CONTENT

LIST OF ABBREVIATIONS

1. ML : Machine Learning
2. XGB : Extreme Gradient Boosting
3. KNN : K-Nearest Neighbor
4. LGBM : Light Gradient Boosting Machine
5. MLP : Multi-Layered Perceptron
6. SVC : Support Vector Classifier
7. SGD : Stochastic Gradient Descent
8. GBM : Gradient Boosting Algorithm
9. RF : Random Forest
10. ANN : Artificial Neural Networks
11. NB : Naïve Bayes
12. NN : Neural Network
13. PIDD : Pima Indians Diabetes Database

LIST OF FIGURES

LIST OF GRAPHS

LIST OF TABLES

ABSTRACT

Diabetes has become a widespread non-communicable disease in modern times, with a prevalence rate of over 60%. The consumption of foods high in carbohydrates, fats, oils, nuts, and whole grains, coupled with factors like alcohol consumption, smoking, rising pollution levels, and high cholesterol, are some of the main reasons why the number of diabetes cases in India is on the rise. Early detection of diabetes can help prevent several severe health conditions that can significantly impact the quality of life for individuals suffering from the disease. As such, a system that can accurately diagnose whether a person has diabetes or not is necessary.

To achieve this, machine learning can be an incredibly useful tool. By analyzing different attributes and utilizing various techniques, it is possible to build a model that can accurately predict diabetes. In the proposed work, the diabetes 2019 dataset obtained from Kaggle, which includes 17 attributes and one target variable, is used to train and test the model. In the dataset there are critical attributes like age, physical activity, family history, blood pressure, body mass index (BMI), and alcohol consumption/smoking habits and so on. A new stack-based ensemble model has been proposed, this model outperforms the conventional machine learning approaches benchmarked by various performance metrics like accuracy, precision, and recall. The proposed approach achieved an impressive accuracy score of 97.70%, significantly better than all other models tested.

CHAPTER-1 INTRODUCTION

1.1 Introduction

Diabetes mellitus, commonly referred to as diabetes, is a metabolic disorder that affects millions of people worldwide [1]. The condition causes the body to produce excessive levels of glucose, which can lead to a host of health complications. After consuming a meal, any carbohydrates ingested are broken down into glucose and absorbed into the bloodstream. When the pancreas detects an increase in blood glucose, it releases insulin to help transport sugar from the blood into cells where it is stored or used for energy. However, when the body fails to produce enough insulin, the insulin produced does not function correctly, or cells do not respond well to insulin, diabetes can occur.

In 2017, it was estimated that 451 million people worldwide were living with diabetes, and this number is expected to rise to over 693 million by 2045 [2]. Shockingly, 57% of these individuals remain undiagnosed, highlighting the need for greater awareness and screening efforts. In India, the prevalence of diabetes and prediabetes is alarmingly high, with one out of six Indian men exhibiting high blood sugar levels, indicating either prediabetes or diabetes. Southern and eastern states, including Kerala, Tamil Nadu, and Andhra Pradesh, report the highest incidences of diabetes, with figures as high as 27%. Indian women exhibit a lower prevalence of diabetes and prediabetes than men, with 13.5% of women and 15.6% of men affected [3].

The complications of diabetes are numerous, with both macrovascular and microvascular complications affecting multiple organ systems. Such complications lead to a higher morbidity and mortality rate, as a result, life expectancy falls and the Indian healthcare system bears a substantial financial burden. Several factors contribute to the rising number of diabetes cases in India, including an increase in the consumption of foods high in carbohydrates, fat, and oils, a lack of fruits, nuts, and whole grains, excessive alcohol consumption, smoking, rising pollution levels, and high levels of cholesterol.

To combat this growing epidemic, greater public awareness, early detection, and management of diabetes are essential. Having a healthy routine that includes exercising regularly, eating a balanced diet, and preventing harmful habits such as consuming cigarettes and too much alcohol will help lower the chance of getting diabetes and its consequences. Additionally, regular screening, early diagnosis, and effective treatment are crucial in reducing the morbidity and mortality associated with diabetes.

Types of Diabetes-

I. Type 1 Diabetes [4]

Type 1 diabetes is an autoimmune disorder in which the body attacks itself, its own insulin-producing cells, located in the pancreas. This can result in a lack of insulin in the body, leading to high blood sugar levels. While type 1 diabetes is often diagnosed in children and young adults, it can occur at any age, and the diagnosis is no longer limited to just this age group.

One common misconception is that all people with type 1 diabetes are thin, but up to 10% of those diagnosed are overweight. This underscores the importance of understanding that diabetes can affect anyone, regardless of body weight or size. Polyuria, or excessive urination, is a hallmark symptom of type 1 diabetes. This is often accompanied by excessive thirst, polydipsia, and hunger. In addition, individuals with type 1 diabetes may experience weight loss, visual abnormalities, and fatigue.

To manage their condition, individuals with type 1 diabetes need to take insulin every day. This can be delivered via injection or through an insulin pump. While this can be challenging, patients of diabetes type 1 can live long and well lives with adequate treatment and care. Diabetes type 1 is a complex condition that requires careful management and attention. With the right approach and access to necessary medical care, people can thrive and live fulfilling lives. It's important to raise awareness of this condition and support those living with it, as they navigate the challenges and complexities of managing their health.

II. Type 2 Diabetes [5]

Diabetes of type 2 is a prevalent persistent illness that impacts a vast number of individuals worldwide. In this condition, the body cannot effectively use insulin to maintain normal blood sugar levels. It's estimated that type 2 diabetes affects between 90-95% of all people with diabetes, making it the most common form of the condition. Type 2 diabetes is a complex condition that can take several years to develop. While it's most commonly diagnosed in adults, it can occur at any age, As a result, it is critical to identify the risk factors linked with the illness. People who are overweight, have a family history of diabetes, lead a sedentary lifestyle, have had gestational diabetes, prediabetes, high blood pressure, high cholesterol, or high triglycerides are more likely to develop type 2 diabetes.

Prevention is key when it comes to type 2 diabetes, and periodic tests might be beneficial. with early detection & management. It's a good idea to get your blood sugar checked at regular intervals, especially if you have risk factors for the condition. Making modifications to your lifestyle, like increasing physical activity, consuming a healthy food, and having an appropriate body mass index can also lower the chance of acquiring type 2 diabetes. Diabetes of the type 2 variety is a complicated disease that affects a great number of people globally. Understanding the risk factors associated with the condition and taking preventative measures can help reduce the likelihood of developing this condition. By staying informed and taking steps to manage our health, we can work towards a healthier future for ourselves and our communities.

III. Gestational Diabetes [6]

Pregnancy is a special time for every woman, filled with excitement and joy. However, hormonal changes can lead to gestational diabetes, diabetes which develops during labour. This syndrome is produced by placental hormones that cause the cells in the body of a pregnant woman less responsive to insulin. As a result, elevated levels of sugar in the blood occur throughout labour, which can have harmful effects on both the mother and the baby. Gestational diabetes is more likely to affect women who are overweight before getting pregnant or who gain excessive weight during pregnancy. During the twenty-fourth and twenty-eighth weeks of pregnancy, routine tests are performed to check for gestational diabetes. This involves an oral glucose tolerance test or a blood sugar test.

Apart from gestational diabetes, there are other less common types of diabetes, such as monogenic diabetes, which is an inherited form of the disease, and diabetes that is linked to cystic fibrosis. However, the most common type of diabetes is type 2, which affects 90-95% of people with diabetes. Over time, elevated levels of sugar in the blood can cause a variety of health issues, including stroke, coronary artery disease, kidney illness, vision difficulties, damage to the nerves, and issues with your feet. It is essential to get your blood sugar levels checked regularly to detect diabetes early on. While there are various tests available to diagnose diabetes, they can be expensive.

Using machine learning algorithms to build a model for the early detection of medical conditions like diabetes is a significant advancement in the field of medicine. The integration of technology in healthcare has brought about tremendous improvements in patient care and treatment outcomes. The development of an early detection system using machine learning can potentially save millions of lives by detecting the onset of the disease before it causes irreversible damage to the body.

The use of an ensemble technique [7, 8, 9] in machine learning can significantly improve the accuracy of the model, making it more reliable and effective in predicting the occurrence of diabetes. Ensemble learning is a powerful technique that combines multiple models to make more accurate predictions. It can be compared to the principle of "two heads are better than one."

Incorporating technology into healthcare not only improves the accuracy of medical diagnoses but also reduces the overall cost of healthcare. With the rise in healthcare costs, people are turning towards more affordable and accessible healthcare options. This includes the use of telemedicine, which enables people to consult with healthcare providers online. The use of machine learning algorithms to diagnose medical conditions like diabetes is another example of how technology is revolutionizing the healthcare industry.

1.2 Problem Statement

Diabetes is a chronic condition which impacts masses of individuals globally and, if left untreated, can lead to major health issues. Living with diabetes can be challenging, and it requires careful monitoring of blood sugar levels, regular medication, and a healthy lifestyle. Many people with diabetes suffer from various complications, such as cardiovascular disease, nerve damage, and kidney problems. Unfortunately, identifying diabetes at an early stage can be difficult, and many patients only find out after experiencing symptoms like excessive thirst and urination, fatigue, and blurry vision. In India, where the prevalence of diabetes is high, access to accurate and timely diagnosis is crucial. The traditional method of visiting a medical diagnostic centre and speaking with a medical professional, can be costly and time-consuming. Moreover, the available diagnostic techniques may not always be accurate, leaving many patients undiagnosed or misdiagnosed.

To address this problem, the aim of the project is to develop a machine-learning system that can predict diabetes at an early stage with a higher accuracy rate. By analyzing various patient features, such as age, body mass index, blood pressure, and family history, the system can determine the correlation between them and build a predictive model using an ensemble technique. This system has the potential to provide a more efficient and accurate diagnosis for patients, enhancing their standard living and minimising the possibility of complications linked with untreated or undiagnosed diabetes.

The main challenge is to find the right combination of features and optimize the model's performance to achieve high accuracy, sensitivity, and specificity while minimizing false positives and false negatives.

1.3 Objectives

- The main objectives of this study are centered around designing, implementing, and evaluating a machine learning model for early diabetes prediction.
- The foremost objective is to enhance the accuracy by using stack ensemble model.
- The results of the ensemble model will then be compared with the existing state-of-the-art literature and, with machine learning algorithms using different metrics.
- By achieving these objectives, the ultimate objective is to enable healthcare providers to identify and intervene early to prevent or manage diabetes more effectively and improve the quality of life for those living with the disease.

1.4 Methodology

The dataset [10] for this project has been sourced from Kaggle and it contains information about 952 patients. There are 17 independent variables that can help determine whether a person is diabetic or not. However, the dataset needs to be pre-processed to achieve better results. In this regard, null values are removed from different columns and replaced with the mean or mode of that column. Some null values are also dropped. Outliers, which are extreme values that could skew the data, are also removed. In addition, some columns are normalized to ensure that they are on a similar scale, and data binning is done on some columns like age, stress, and BMI. Binning is a method of organizing several more or less continuous values into fewer "bins". For example, if a person has information about a group of people, they might want to divide their ages into fewer age ranges. This makes it easier to analyze the data. The correlation between different columns is also analyzed to gather information on how one column is related to another.

The process of developing a machine learning model involves several steps, including data preparation, algorithm selection, model training, and testing. By using the right techniques and performance metrics, the accuracy of the model can be significantly improved, making it more effective in solving real-world problems like predicting diabetes. One of the most important phases is to separate the data into a testing and training dataset, which allows the effectiveness of the model on unknown data to be evaluated. Typically, the ratio of splits is 80:20, with training data as 80% and testing as 20%.

In the instance of diabetes prediction, eleven machine learning algorithms are implemented. These algorithms are trained on the training dataset and tested on the testing dataset. The performance of the models is compared using several measures such as f1-score, recall, precision and accuracy to determine which algorithm performs better in predicting whether a person is diabetic or not.

To further enhance the model's accuracy, the stacking ensemble technique is applied. This technique involves combining the predictions made by multiple base learner models with a meta-learner to create a final model with more accurate predictions. The stacking technique protects the strengths of different models, making them more effective in addressing classification and regression problems.

We have applied two-class stacking with different combinations of random forest classifier, xgboost, decision tree classifier, extra tree classifier, and light gradient boost machine classifier as the base learner and meta learner. The meta learner takes the outputs of the base models as inputs and makes a final decision based on their predictions. After testing the ensemble on testing data, we achieved an impressive accuracy of 97.70%. This result shows the power of ensemble techniques in improving the accuracy of machine learning models for predicting diabetes.

1.5 Organization

There are 5 sections in the project report.

- Chapter 1 aims to provide a concise introduction to diabetes and explain the problem statement, emphasizing the efficacy of the proposed application.
- Chapter 2 presents the project's literature review, citing relevant sources.
- Chapter 3 covers dataset details, data preprocessing, model descriptions, and the proposed framework.
- Chapter 4 delves into the experiments and results.
- Chapter 5 concludes the project by discussing its overall implications and future possibilities.

CHAPTER-2 LITERATURE SURVEY

In this section, we have reviewed around some researches and studied how the researchers have used the different methods using different dataset to get an accurate machine learning model for diabetic prediction. Over the years various techniques were developed by the researchers, some of them are discussed these. Also, the approaches and datasets used by different researchers with their results are shown in table 1.

Tigga et al. [10] gathered a dataset of 952 persons aged 18 and above, including characteristics such as health, lifestyle, and family history. They used decision tree, naïve bayes, support vector machine, k- nearest neighbour, random forest, logistic regression models and attained a highest accuracy of 94.10% with Random Forest.

In [11], The authors used machine learning classification techniques such as Decision Tree, Naive Bayes, and SVM to the Pima Indians Diabetes Dataset (PIDD) from the UCI machine Learning library. They concentrated on diabetic pregnant women and utilized the WEKA tool to execute the experiment. The author assessed metrics such as Precision, Recall, Accuracy, F-measure, and ROC for comparing the different models. The Naive Bayes algorithm is regarded as the best-supervised machine learning approach in the experiment since it produces the greatest results with an accuracy of 76.30%.

The National Institute of Diabetes and Digestive and Kidney Diseases Dataset, which is accessible at the UCI ML Repository, was utilized by Alam et al. [2]. On the dataset, K-means clustering classification, RF & the ANN models were used sequentially. The accuracy and the AUROC curve were used to evaluate the findings. With an accuracy of 75.7 percent, ANN surpassed the other two models.

From 2011 to 2017, the authors of [12] used physical examination data from the EMR of the Luzhou Municipal Health Commission in China. They evaluated the best prediction model using random forests (RF), XGBoost, and logistic regression (LR). They developed an online diabetes risk assessment method to estimate the risk of diabetes.

Azrar et al. [13] used the Pima dataset, which contains records of females over the age of 21 who live in Phoenix, Arizona, USA. K-Nearest Neighbor (KNN), Decision Trees (DTs), and Naive Bayes are among the algorithms used, with Decision Trees having the best accuracy of 75.65%.

In [14], Singh et. al used function-based multilayer perceptron (MLP), probabilistic-based NB, and DT-based random forests (RF) on PIDD with 10-fold cross-validation and split of 66% of the training dataset, also using training dataset as the test dataset.

Ayon et al. [15] used the same dataset of Pima Indians Diabetes and applied deep neural networks and obtained the greatest results with the hidden layer being four and the number of neurons in each hidden layer being twelve, sixteen, sixteen and fourteen. They achieved the highest accuracy 98.35% with Five-fold cross-validation.

In [16] the authors used the dataset from documentation of the Association of diabetic's city of Urmia which has 1004 samples with nine attributes, and applied the artificial neural networks which got 87.3% accuracy and the average error function was 0.01 with neural networks.

Khanam et al. [17] used the same PID dataset available at the UCI machine learning repository. Then applied different ml algorithms i.e. decision tree, Knn, random forests, NB, ada boost, linear regression, SVM and NN. 88.6% accuracy was achieved with NN model with two hidden layers and various epochs.

The diabetic Retinopathy dataset is used by Reddy et al. [18]. They applied the min-max normalization method and then used the adaboost rf classifier, logistic regression classifier, knn, decision tree classifier machine learning models based on these they build a machine learning ensemble model with 84% accuracy.

In [19] Sarwar et al. developed an AI-based ensemble tool for the detection of diabetes and analyzed its efficacy to that of other machine learning approaches. They used 400 people dataset and ensembled naive bayes, k-nearest neighbor (knn), support vector machine (svm), and artificial neural network (ann), models got an accuracy of 98.60%.

Singh et al. [20] build an ensemble framework, ensembling different ml algorithms consisting a NN, DT, SVM, RF, and XGBoost which achieved an accuracy of 95%. Accuracy, Gini Index, precision, specificity, sensitivity, area under the convex hull, area under the curve, minimum weighted coefficient, and minimal error rate were all used to measure performance.

Gradient boosting, logistic regression, DT, ETC, lgbm (light gradient boosting machine), xgboost, and random forest machine learning models were used by Ahamed et al. [21] on the Pima Indian dataset and 95.20% highest accuracy was attained by lgbm.

A soft voting ensemble classifier was proposed by Kumari et al. [22] which achieved an accuracy of 79.04% which includes cat boost, bagging, support vector machine, Adaboost, xgboost, NB, logistic regression, gradient boost, RF machine learning models on the Pima Indians diabetes database.

An artificial back propagation scaled conjugate gradient neural network (ABP-SCGNN) was used to train the ANN model was proposed by Bukhari et al. [23] on the PID dataset which attained an accuracy of 93%.

In [24] Hasan et al. framed an ensemble model consisting of K-NN, Random Forest, AdaBoost, Decision Trees, MLP, XGBoost, and Naive Bayes algorithms on the PID dataset. They attained the highest AUC value of 0.95.

Ensemble approaches like as bagging and boosting were used in [25] by Nai-Arun et al. The experiment used hospital data from the Sawan Pracharak Regional Hospital in Thailand, which included 48,763 patients. The attributes were sorted using the ratio of gain selection of features methodology, and the Naive Bayes, K-Nearest Neighbours, and Decision Tree algorithms were used in the following phases. The boosting and bagging ensemble approaches relied on these algorithms as basis classifiers. The bagging strategy was shown to be more successful than the boosting method, with an accuracy of 95.312%. It also surpassed every one of the base classifiers. Future studies might look into the stacking approach, according to the author.

Swapna et al. [26] used Deep Learning techniques to predict diabetes, training a Multilayer Feed-Forward Neural Network with the back-propagation algorithm. They normalised the PIMA Indian dataset before processing it to guarantee numerical stability, reaching a noteworthy accuracy of 82%. In another work, the researchers trained two models using CNN and CNN-LSTM on an Electrocardiograms data set with 142,000 samples and eight variables. The CNN model had an accuracy of 93.6% with five-fold cross-validation, whereas the CNN-LSTM model had an accuracy of 95.1%.

Chen et al. [27] examined the performance of the J48, KNN, and LR algorithms on the diabetes dataset, concluding that the J48 method had the greatest accuracy rate of 78.27%. Based on their results, another researcher [28] created a web-based application for identifying diabetes mellitus that reached an accuracy rate of 80%. The researchers tested numerous prediction algorithms in this study, including DT, NN, (NB), logistic regression, and random forest, as well as ensemble techniques. They discovered that the RF approach has the highest accuracy and ROC score of 75.558% and 0.912.

S. Sivaranjani et al. [29] used the Pima dataset to train two machine learning algorithms: RF & SVM in their work. They pre-processed the data first, then used step forward and step backward feature selection strategies to compare their efficacy. Furthermore, Principle Component Analysis (PCA) was used to minimise the dimensionality of the dataset; however, due to the magnitude of the dataset, this had little effect on the models' performance. Finally, they compared the performance of the two models using four features and discovered that the RF with step backward feature removal had the greatest specificity (82%), sensitivity (83%) & accuracy (83%).

Table 1: List of existing approaches and dataset used for diabetic prediction

S. No.	**Author (s)**	**Approach**	**Dataset**	**Results**
1	Tigga et al. (2020) [10]	Logistic Regression, SVM, KNN, DT RF, NB	Tigga Dataset on Kaggle	94.10% with Random Forest

2	Sisodia et al. (2018) [11]	Decision Tree, Naive Bayes, and SVM machine learning classification algorithms	PIDD from UCI machine Learning repository.	76.30% Accuracy with Naive Bayes.
3	Alam et al. (2019) [2]	K-means clustering, random forest (RF), and Artificial neural network (ANN) models	National Institute of Diabetes and Digestive and Kidney Diseases Dataset	75.7% Accuracy with ANN
4	Yang et al. (2021) [12]	XGBoost, random forests (RF) and logistic regression (LR)	Physical examination data from the EMR of Luzhou Municipal Health Commission in China	0.8768 AUC with XGBoost
5	Azrar et al. (2018) [13]	K-Nearest Neighbour (KNN), Decision Trees (DTs), Naïve Bayes	Pima Indians Diabetes Dataset (PIDD), (UCI)	75.65% Accuracy with Decision Trees

6	Singh et al. (2017) [14]	Multilayer perceptron (MLP), probabilistic-based NB, and decision tree-based random forests (RF)	Pima Indians diabetes dataset (PIDD), (UCI)	84.93% with Random Forests
7	El Jerjawi et al. (2018) [16]	Artificial Neural Network	From documentation of the Association of diabetic’s city of Urmia	87.3% with ANN
8	Khanam et al. (2021) [17]	DT, K-Nearest Neighbour, Random forests, NB, Ada Boost, Linear Regression Neural Network (NN) and Support Vector Machine	Pima Indian Diabetes (PID) Dataset, (UCI)	88.6% with NN with two hidden layers
9	Reddy et al. (2020) [18]	Ensemble model including Random Forest, Decision Tree, Adaboost, KNN, Logistic Regression classifiers	Diabetic Retinopathy dataset	84% with ensemble model

10	Hasan et al. (2020) [24]	Ensemble Framework including K-NN, Random Forest, AdaBoost, Decision Trees, MLP, XGBoost, and Naive Bayes	Pima Indian Diabetes (PID) Dataset, (UCI)	0.95 AUC with AB & XB
11	Singh et al. (2021) [20]	Ensemble model with NN, DT, SVM, RF and XGBoost	Pima Indian Diabetes (PID) Dataset, (UCI)	95% with ensemble model
12	Ahamed et al. (2022) [21]	Logistic Regression, RF, DT, Light Gradient Boosting Machine (LGBM), Gradient Boosting, XGBoost	Pima Indian Diabetes (PID) Dataset, (UCI)	95.20% with LGBM
13	Kumari et al. (2021) [22]	Ensemble soft voting classifier	Pima Indian Diabetes (PID) Dataset, (UCI)	79.04% with ensemble classifier
14	Bukhari et al. (2021) [23]	ANN model trained using an artificial back propagation scaled conjugate	Pima Indian Diabetes (PID) Dataset, (UCI)	93% with ABP-SCGNN

		gradient neural network		
15	Sarwar et al. (2018) [19]	Ensemble method with ANN, SVM, KNN, Naive Bayes	Their own dataset of 400 people	98.60% with ensemble method
16	Gollapalli et al. (2022) [30]	Stacking (combined K-NN, Bagging DT, & Bagging K-NN, with K-NN meta-classifier	Saudi Arabian hospital dataset	94.48% with Stacking
17	Latif et al. (2021) [31]	Max Voting, and Stacking ensemble technique	PIMA Indians Diabetes dataset, Vanderbilt Dataset	78% for Dataset 1 and 93% for Dataset 2 with max voting
18	Husain et al. (2018) [32]	Ensemble model using majority voting technique (Using and Gradient Boosting, Random Forests, K-Nearest Neighbor, and Logistic Regression)	NHANES 2013-14	0.75 AUC with Majority voting

19	Mahesh et al. (2022) [33]	Blended ensemble learning (using random forest (RF), K-nearest neighbors (KNN), support vector machine (SVM), decision tree (DT) classifier, & logistic regression (LR))	Kaggle dataset	97.11% with Ensemble technique
20	Ali et al. (2014) [34]	Boosting ensemble technique (using Random committee classifier & AdaboostM1 algorithm)	Diabetes dataset of 100 patients records from a local hospital	81.0% with boosting
21	Dutta et al. (2022) [35]	Ensemble technique (using LightGBM (LGB), XGBoost (XGB), Decision Tree (DT), Random Forest (RF) and Naive Bayes (NB))	South Asian diabetes dataset	73.50% with ensemble technique
22	Saxena et al. (2022) [36]	Random Forest, Decision tree, KNN, MLP	PIMA	79.80% with RF

23	Maniruzzaman et al. (2020) [37]	AdaBoost, random forest, Naive bayes & Decision tree	National Health and Nutrition Examination Survey	94.25% with random forest
24	Laila et al. (2022) [38]	RF, Bagging, AdaBoost	PIDD	97% with RF

CHAPTER-3 SYSTEM DEVELOPMENT

In this chapter, you will find valuable information regarding the dataset used for the study, which will help you gain a better understanding of the research outcomes. The dataset used in this study was carefully selected and analyzed, and the patterns identified were visually presented in the form of graphs and tables. These visual representations can help you identify key trends and patterns in the data, making it easier for you to draw insightful conclusions.

Apart from the dataset, the proposed model used in the study is also presented in this chapter. The proposed model is the result of extensive research and analysis, and it has been carefully designed to address the research question at hand. You will get a detailed insight into the model and how it works.

Furthermore, the chapter also explains the algorithms that led to the development of the proposed model. This will give you an idea of the research methodology and the steps taken to ensure the accuracy and validity of the results. Also, the architecture of the proposed stack ensemble method is shown in figure 1.

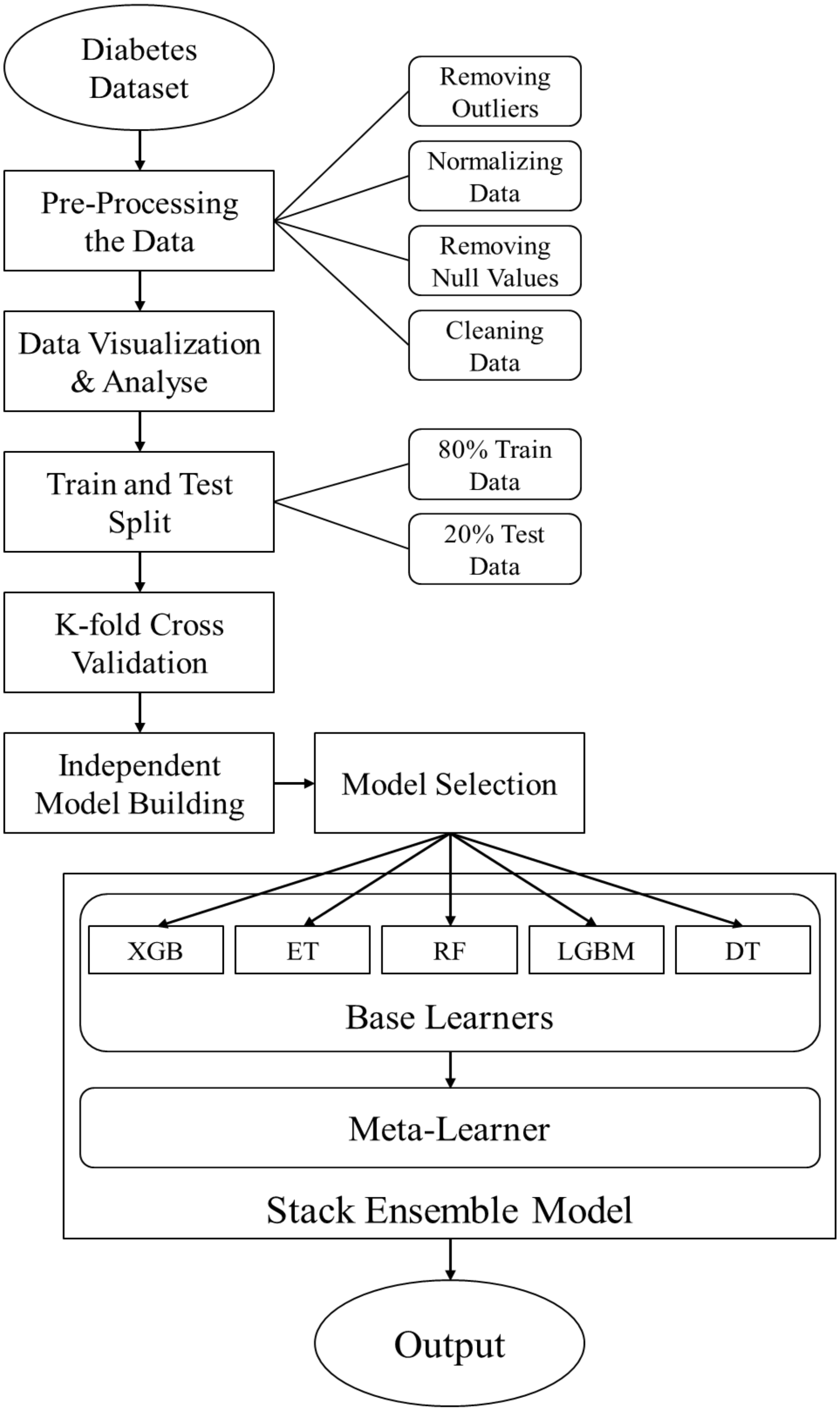


Figure 1: The architecture of the recommended stack ensemble method

3.1 Dataset description

The source of the dataset is Kaggle, a platform for data science competitions. The particular dataset being referred to is the Diabetes dataset from the year 2019. It was curated by Dr. Shruti Garg and Neha Prerna Tigga for the sole purpose of research and non-commercial use. The dataset contains 952 cases, each with 17 independent factors that predict results and a binary dependent variable that indicates the presence or absence of Diabetes. Among the instances, there are 580 males and 372 females, providing a significant representation of both genders. The dataset includes several real-world factors that can be leveraged to accurately identify whether a person is suffering from diabetes or not.

Table 2: Description of Attributes in the Dataset

Serial Number	**Attributes**	**Type**
1	Age	60 and above 50-59 40-49 Less than 40
2.	Gender	Male Female
3.	Family Diabetes	Yes No
4.	High BP	Yes No
5.	Stress	Always Very Often Sometimes

		Not at all
6.	BMI	Numerical value
7.	Bp level	Low Normal High
8.	Regular Medicine	Yes No
9.	Sleep	Numerical value
10.	Sound Sleep	Numerical Value
11.	Alcohol	Yes No
12.	Junk Food	Occasionally Often Very Often
13.	Physically Active	One hour or more More than half an hour Less than half an hour None
14.	Smoking	Yes No
15.	Pregnancies	0 1 2 3 4
16.	Pdiabetes	Yes No
17.	Urination Frequency	Not Much Quite Often

18.	Diabetes	Yes No

In this dataset, various attributes are included to determine the likelihood of a person being diabetic. The first attribute, age, is binned into four categories to make it easier to analyze: less than 40, 40-49, 50-59, and 90 and above. This is important as age plays a significant role in determining a person's susceptibility to diabetes. For example, individuals over the age of 60 are at a greater chance of acquiring diabetes than those under the age of 60.

Gender is another attribute that is included in the dataset. It is divided into two categories: male and female. This is important as research has shown that men and women may have different risks of developing diabetes. For example, women with polycystic ovary syndrome (PCOS) are more likely to develop diabetes than those without the condition.

Family diabetes is another attribute that is included in the dataset. This attribute has two columns: yes and no. It explains whether there is a past history of diabetes in the family, which is important as genetics plays a significant role in determining a person's susceptibility to diabetes.

High blood pressure (BP) is another attribute included in the dataset. It is marked by yes or no and defines whether the person has high blood pressure or not. This is important as high BP is threat for developing diabetes. Research has shown that individuals with People with elevated levels of blood pressure are more prone to acquire diabetes than those who do not have high blood pressure.

Physical activity is another attribute included in the dataset. It defines how long a person plays outside and is physically active. It is also divided into various categories: one hour or more, more than half an hour, less than half an hour, and none. This is important as physical activity plays a significant role in preventing diabetes. Physically active people are less likely to get diabetes than inactive people, according to research.

Body mass index (BMI) is another attribute included in the dataset. It is a numerical value that is calculated based on the height and weight of the person and varies among different people. This is important as BMI is a significant threat for developing diabetes. Individuals with a higher BMI have a greater probability to acquire diabetes than those with a lower BMI, according to research.

Smoking and drinking are other attributes included in the dataset. These attributes are characterized by yes or no and define whether a person smokes or drinks. This is important as smoking and drinking are significant risk factors for developing diabetes.

Sleep and sound sleep are other attributes included in the dataset. These attributes are numerical values that define how much sleep a person gets in the daytime and how long a person sleeps without any disturbance, respectively. This is important as research has shown that individuals who do not get enough sleep or who experience disturbed sleep are far more inclined to diabetes than those who get sufficient and undisturbed sleep.

Regular medicine is another attribute included in the dataset. It has a yes and no category and defines whether a person is on a medicine and if he/she takes that medicine regularly. This is important as some medicines can increase the risk of developing diabetes. For example, Certain steroids can raise insulin levels and raise the risk of diabetes.

Junk food is another attribute included in the dataset. It defines how often a person eats junk food in his daily lifestyle and is divided into three categories:

occasionally, often, and very often. This is important as research has shown that a diet high in junk food is a significant risk factor for developing diabetes.

Stress is another aspect of our lives that can greatly affect our health. The stress column in this dataset is a measure of how frequently a person experiences stressful situations. Stress can manifest in various ways, and can have negative effects on our physical and mental wellbeing if not managed properly. It's categorized into four levels: not at all, sometimes, very often, and always.

One important attribute for determining a person's health condition is their blood pressure level. The BP level column in this dataset is divided into three categories: low, normal, and high. High blood pressure can be an indicator of diabetes, so monitoring this measurement can be crucial in identifying potential risks for the disease.

For women, the number of pregnancies they've had can also impact their health status. The pregnancies column in this dataset is divided into five categories, ranging from 0 to 4. This information can be helpful in assessing a woman's risk for gestational diabetes, a type of diabetes that can develop during pregnancy.

The pdiabetes column in this dataset is a binary value, with "yes" indicating that a person has diabetes and "no" indicating that they do not. This information can be crucial in identifying potential risks for the disease and taking preventative measures.

The urination frequency column in this dataset measures how often a person urinates throughout the day, and is divided into two categories: "not much" and "quite often." This can be a useful indicator for diabetes, as increased urination frequency can be a symptom of the disease.

Finally, the target column in this dataset predicts whether a person has diabetes or not, and is represented by the binary values "yes" and "no." By analyzing the other attributes in the dataset, healthcare professionals can use this column to make informed predictions about a person's health status and take appropriate measures to manage or prevent diabetes.

3.2 Data preprocessing and data visualization

The dataset we are working with contains information about various health indicators such as BMI, pregnancy, diabetes, and sleep. However, we noticed that some of these columns have missing data. This is not uncommon as missing data is a common issue in datasets. To handle this, we employed a few techniques to fill in the gaps.

During our data analysis, we encountered some outliers, which are data points that are vastly different from the other values in their respective column. These outliers can heavily affect the accuracy of statistical tests and lead to increased error variance. They can distort the overall results of our analysis and make any projections based on them inaccurate. Therefore, we took steps to identify and remove any outliers present in the data.

For instance, in a survey regarding the impact of stress on people's health, we noticed that some participants reported extremely high levels of stress that were significantly different from the average level. By removing these outliers, we could obtain a more accurate representation of the dataset, and this helped to improve the quality of our analysis.

Additionally, we found that some columns, such as age, stress, and physical activity, had categorical attributes that needed to be converted into numerical attributes for further analysis. To convert these categorical attributes into numerical attributes, we used a technique called one-hot encoding. This technique involves creating dummy variables for each category, where the variables' values are either 0 or 1, indicating the absence or presence of a specific category in a row. This way, we could transform the categorical variables into a format that can be easily analyzed and compared, which helps to improve the accuracy of our analysis.

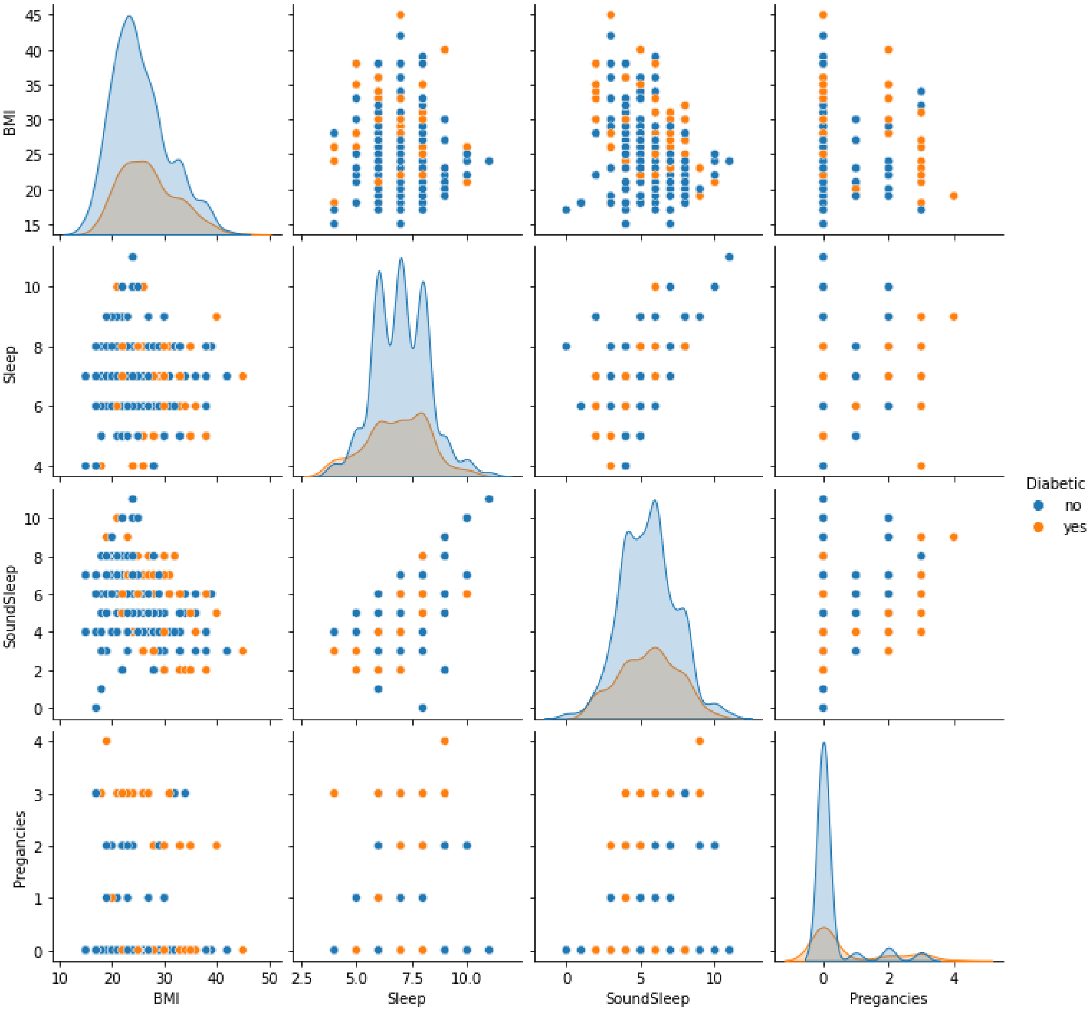
For the Pregnancy column, we filled in the missing values with the mode value of the column. This is because the mode represents the most commonly occurring value in a column, so it was the best option to fill in the missing data. For the other columns, we used the mean value to fill in the missing data. While there are other methods to fill in missing data, using the mean is a simple and effective way to estimate the missing values.

For other columns like gender, family history of diabetes, smoking, and alcohol consumption, we also converted these categorical attributes into numerical attributes. This is because numerical data is easier to work with and analyze.

Finally, we normalized the values of the BMI and sleep columns to ensure that they were on the same scale as the other columns. This helps to improve the accuracy of our analysis and makes it easier to compare results.

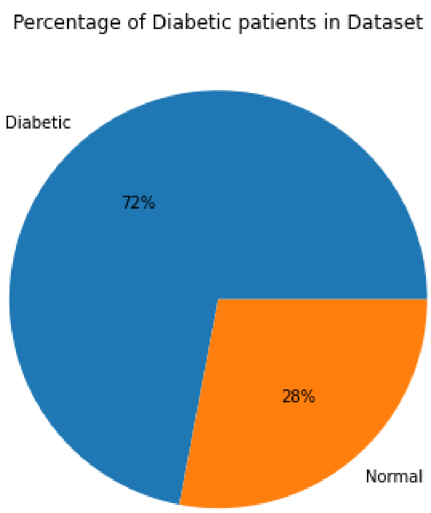
In order to better understand our research findings on diabetes, we decided to create various graphs to visually represent the data. These visuals can be helpful in identifying patterns and trends that may not be immediately apparent from just looking at the raw data.

One of the graphs we created is a Pair plot that displays the relationship between BMI, sleep, sound sleep, and pregnancies shown in Graph 1. This graph helps us understand how these factors are related to one another and how they may impact diabetes.



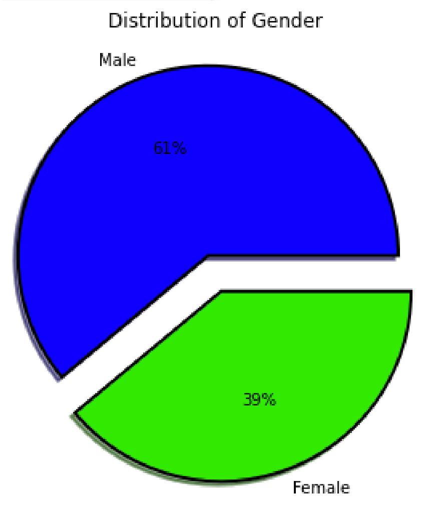
Graph 1: Pair plot of BMI, Sleep, Sound sleep and Pregnancies

Another graph we created is a Pie Chart that shows the distribution of our target variable, which is whether or not a person is diabetic, shown in graph 2. This helps us understand the prevalence of diabetes in our dataset and how it may be influenced by other factors.



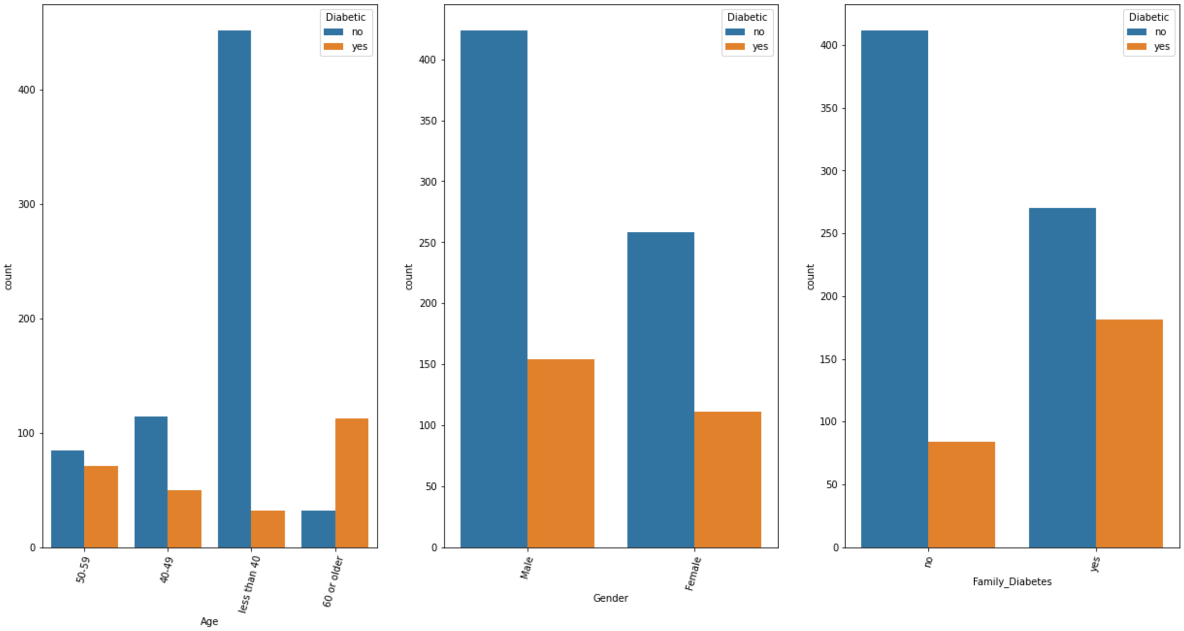
Graph 2: Pie Chart of the Target variable i.e. Diabetic or not

We also created a Pie Chart that shows the distribution of sex in our dataset, shown in graph 3. This helps us understand if there are any differences in health outcomes based on gender.



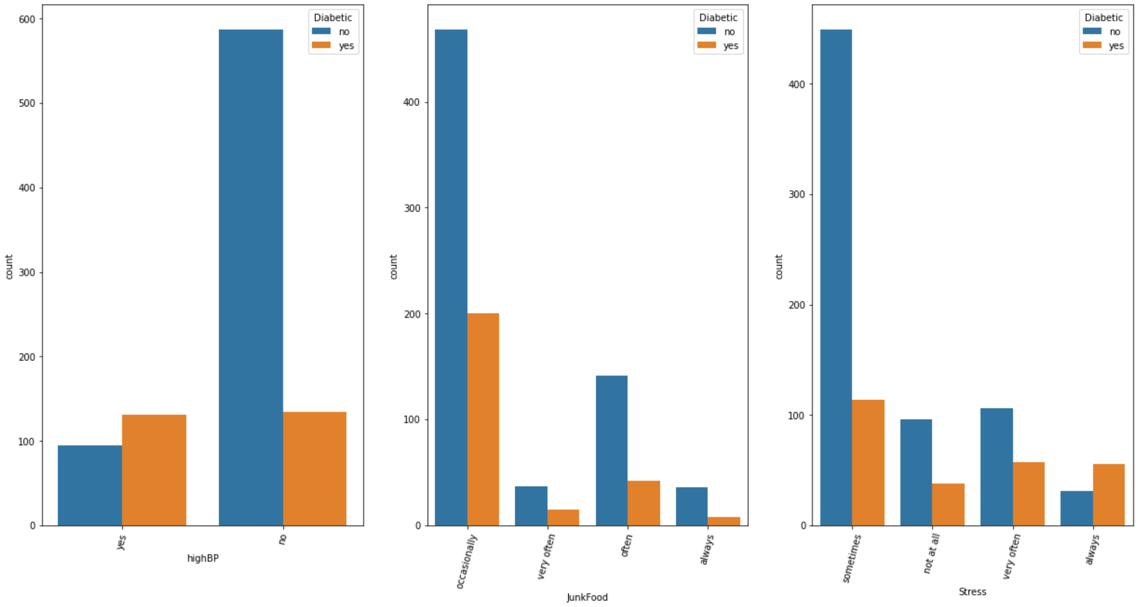
Graph 3: Pie chart of distribution of sex in the dataset

To further explore the relationship between our target variable and various other factors, graph 4 that shows the distribution of age, gender, and family diabetes with respect to our target variable. This helps us understand how these factors may influence the likelihood of developing diabetes.



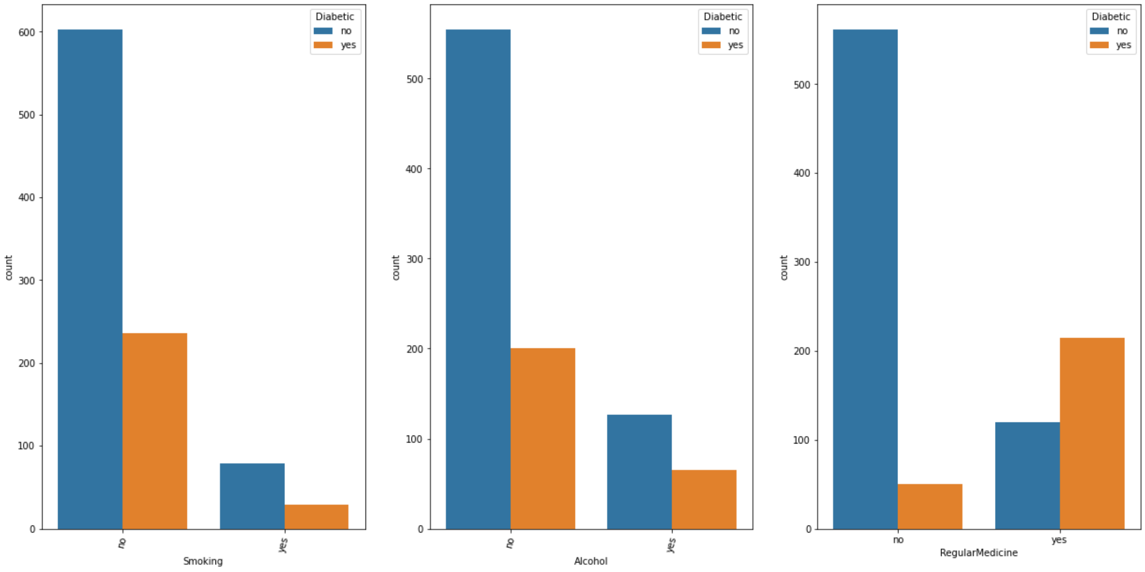
Graph 4: Count of Age, Gender and Family Diabetes with respect to target variable

We also created a Count graph that displays the distribution of high blood pressure, junk food consumption, and stress levels with respect to our target variable, graph 5 shows that. This graph helps us understand how these lifestyle factors may contribute to the development of diabetes.



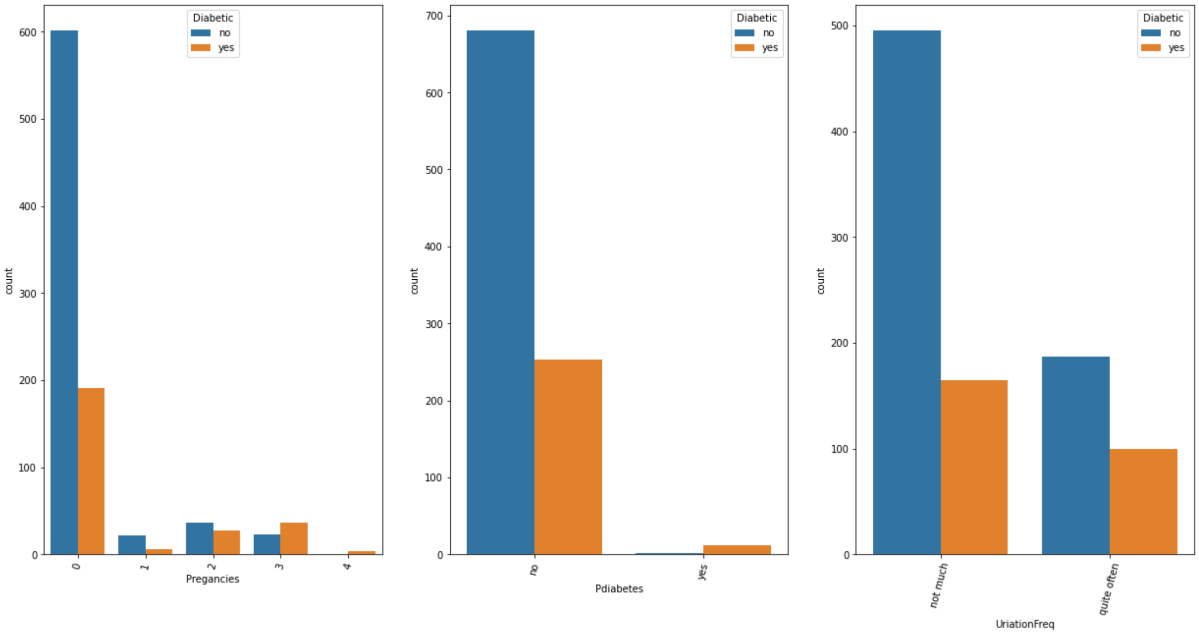
Graph 5: Count of High BP, Junk food and Stress with respect to target variable

Additionally, graph 6 shows the distribution of smoking, alcohol consumption, and regular medication use with respect to our target variable. This graph helps us understand how these factors may impact our overall health and well-being.

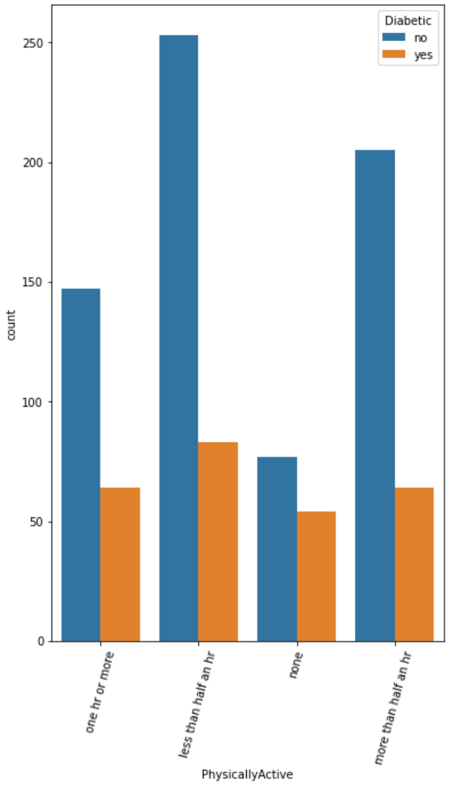


Graph 6: Count of Smoking, Alcohol and Regular Medicine with respect to target variable

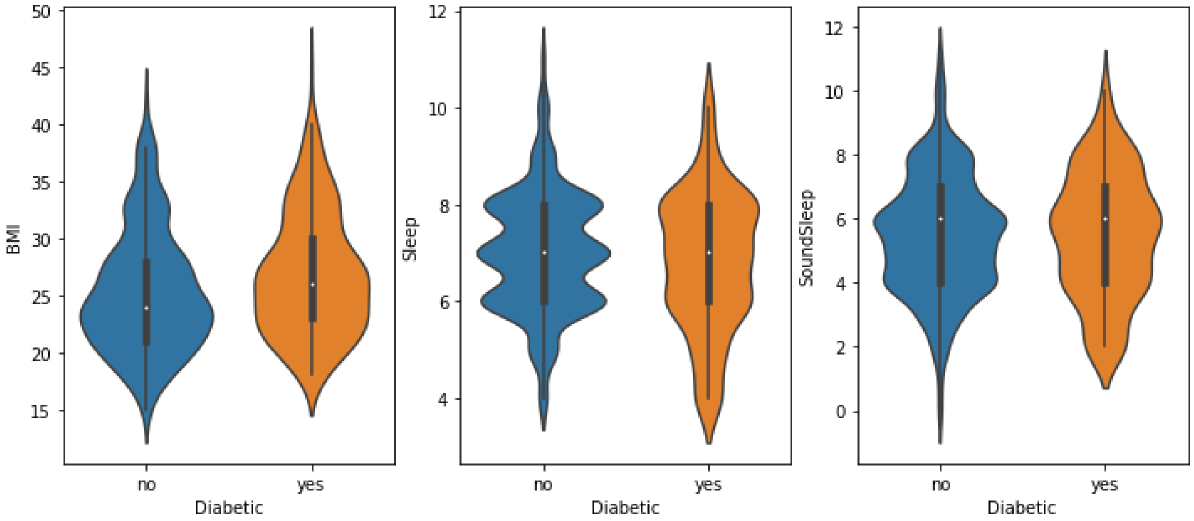
Graph 7 displays the relationship between pregnancies, Pdiabeties and urination frequency, and the likelihood of developing diabetes. This helps us understand how pregnancy and urinary health may impact a person's risk of developing diabetes. And count of Physical activity in Relation to the target variable is shown in graph 8.



Graph 7: Count of Pregnancies, Pdiabeties and Urination frequency in relation to the target variable



Graph 8: Count of Physical activity in relation to the target variable



Graph 9: Violin plot with BMI, Sleep and sound sleep with respect to diabetes

Lastly, we created a Violin plot that displays the relationship between BMI, sleep, and sound sleep with respect to diabetes shown in graph 9. This graph helps us understand how these factors may impact a person's risk of developing diabetes and can inform our recommendations for promoting healthy behaviors.

3.3 Brief description of Machine Learning Classification Techniques

Here, we have delved into a wide range of machine learning classification algorithms that are being utilized for the recommended model. These algorithms form the backbone of our model, and it's important to consider all the available options before deciding on the best approach.

In order to find the optimal solution, we experimented with various classifier models before finally settling on the top performing ones. In total, we trained 11 distinct classifiers on our training data set, each with its own unique strengths and weaknesses.

3.3.1 Random Forest [39]

RF is a popular ml algorithm that is often performs classification and regression tasks using this algorithm. It belongs to the family of ensemble methods, which means that it combines multiple weaker models to create a stronger and more accurate final model.

The basic idea behind Random Forest is to build a large number of decision trees on different subsets of the training data, and then combine their predictions to make the final prediction. Each tree is constructed using a random subset of the features and a random subset of the training samples, which helps to reduce overfitting and increase the generalization performance of the model.

Throughout the estimation phase, every tree in the forest guesses the target variable independently, and the final forecast is generated by taking the majority voting of all the trees. This method has been demonstrated to be quite successful in improving prediction accuracy, particularly in extremely large data sets with complicated connections between characteristics and the variable being targeted.

Random Forest has the benefit of being able to handle both quantitative and qualitative information, as well as automatically handling values that are missing and outliers. It's also relatively easy to use and requires minimal hyper parameter tuning, making it a popular choice for both beginners and experienced machine learning practitioners.

As a whole, Random Forest is a strong and adaptable technique that may be applied to a variety of the two methods applications & is well-suited to handling complex and high-dimensional data sets.

3.3.2 Extra Tree classifier [40]

The Extra Tree classifier is another popular ml algorithm falls in the group of ensemble methods. However, unlike Random Forest, Extra Trees builds decision trees using random thresholds for each feature instead of finding the best split point.

The main idea behind the Extra Tree classifier is to create a huge number of random DT using various subsets of the training data and different subsets of the features. During the training phase, the algorithm randomly selects a threshold for each feature, and then uses these thresholds to split the data into different branches. This process is repeated for each tree in the forest, resulting in a diverse set of decision trees.

During the prediction phase, the Extra Tree classifier aggregates the results of all the individual trees to make a final prediction. This aggregation can be done through different methods, such as taking the majority vote or computing the mean of the expected values.

One of the advantages of the Extra Tree classifier is that it can be faster than other ensemble methods, such as Random Forest, due to its simpler splitting strategy. It also has the ability to handle noisy and irrelevant features more effectively, since the random splitting strategy makes the algorithm less sensitive to the noise and irrelevant features in the data.

However, one potential drawback of the Extra Tree classifier is that it may be more prone to overfitting, especially if the number of trees in the ensemble is too high. This can be mitigated by tuning the hyperparameters, such as the number of trees and the maximum depth of each tree, to optimize the performance of the model.

3.3.3 Decision Tree classifier [41]

A DT classifier is a ml algorithm to categorize data, creates a model in the form of a tree structure. It operates by recursively dividing the data into subsets based on feature values, with the goal of producing subsets that are as pure as feasible with regard to the target variable. The tree is constructed by picking the appropriate feature to partition the data at each node based on some information gained or impurities reducing criteria. This procedure is repeated until a condition, such as the smallest amount of collections in a leaf node or a maximum depth of the tree, is fulfilled.

Once the tree is built, new data can be classified by traversing the tree from the root node to a leaf node, where the prediction is made based on the majority class of the samples in that node.

Decision Trees are popular owing to their comprehensibility as the final model is simple to see and comprehend. They can handle qualitative as well as quantitative data and are relatively resistant to outliers and missing values. However, they are prone to overfitting, especially when the tree is deep, and could not hold up well to fresh datasets. Pruning and ensembling techniques can assist to overcome these challenges.

3.3.4 Light Gradient Boosting Machine Classifier [42]

A LightGBM classifier is a type of machine learning model used for classification tasks. It is a member of the algorithm for gradient boosting family and is noted for its rapid and efficient performance.

LightGBM works by iteratively building a series of decision trees, where each subsequent tree is trained to correct the errors of the previous tree. During training, LightGBM uses a technique called gradient-based one-side sampling (GOSS) to selectively sample data instances based on their gradient values, which helps to speed up training while maintaining accuracy.

The LightGBM classifier is known for its ability to handle large datasets with high dimensionality and imbalanced class distributions. It also supports various loss functions, including binary cross-entropy, multi-class cross-entropy, and AUC-based binary classification.

3.3.5 XG Boost [43]

The Extreme Gradient Boosting (XGBoost) ml technique is widely used for classification & regression and ranking applications. It is a gradient boosting implementation, which is an approach to ensemble learning that constructs a model step by step by minimizing the model's loss function using gradient descent.

XGBoost works by creating a set of decision trees, where each tree tries to correct the errors of the previous tree. During the training phase, the algorithm creates a set of weak learners, each of which predicts the output variable. These weak learners are then combined to form a final prediction that is more accurate than any individual weak learner. XGBoost has become popular due to its high accuracy, efficiency, and flexibility. It can handle missing data, imbalanced classes, and can work with a variety of data types, including numerical, categorical, and text data. It also provides built-in regularization techniques to prevent overfitting and improve generalization.

XGBoost has been used in various domains, including finance, healthcare, and online advertising, where it has achieved state-of-the-art results in many benchmark datasets.

3.3.6 Gradient Boost [44]

Gradient Boosting is a ml approach that may be utilised for classification as well as regression. An ensemble approach which brings together several weak approaches to form a powerful prediction.

The basic idea behind gradient boosting is to sequentially add new models to the ensemble, with each model trained to correct the errors of the previous models. The algorithm works by first fitting an initial model to the data, and then iteratively fitting new models to the residuals of the previous models.

At each iteration, the algorithm determines the direction in which the new model should be fitted, by calculating the negative gradient of a loss function. The loss function measures the difference between the predicted values and the actual values of the target variable, and the negative gradient points in the direction of steepest descent of the loss function.

The fresh approach is then taught to forecast the residuals of the prior models, which are the discrepancies among the target variable's actual and predicted values.

The new model's predictions are added to the prior models' predictions, and the procedure is continued until a halting requirement, such as a maximum number of iterations or a minimum improvement in the loss function, is reached.

Gradient boosting has been shown to be highly effective in a wide range of applications, including prediction, classification, and feature selection. However, it is extremely costly and requires cautious parameter tweaking to avoid overfitting.

3.3.7 K-Nearest Neighbors [45]

KNN is a supervised ml algorithm applied for both the classification and regression tasks. It is a non-parametric method, indicating that it makes no presumptions regarding data dispersion.

The KNN algorithm predicts the class or value of a data point by locating its K closest neighbours and uses most of the category or median value of those neighbouring to forecast the category or values of any given data. The nearest neighbour distance metric might be Euclidean, Manhattan, or any other distance metric.

One important hyperparameter of the KNN algorithm is the value of K, which determines the number of nearest neighbors to consider. A larger value of K makes the algorithm more robust to noise but can result in lower accuracy, while a smaller value of K can result in higher accuracy but can be more sensitive to noise.

3.3.8 Multi-Layered Perceptron [46]

A Multi-Layer Perceptron (MLP) is a type of artificial neural network (ANN) that consists of multiple layers of interconnected nodes, also known as neurons. Each neuron gets input from the layer prior to it, computes it, and outputs an outcome that is transmitted on to the next layer.

The input layer of an MLP consists of the input data, while the output layer produces the final output of the network. The layers between the input and output layers are known as hidden layers, and they perform intermediate computations on the input data to transform it into a useful output. Each neuron in an MLP is typically represented by a mathematical function called an activation function, which is applied to the weighted sum of its inputs to produce its output. The weights are learned through a process called backpropagation, which involves iteratively adjusting the weights based on the error between the network's predicted output and the true output.

MLPs are widely utilised in ml for a wide range of tasks including regression, classification, and finding patterns. They've been used effectively in a variety of fields, including image and audio recognition, processing natural languages, and prediction of finances.

3.3.9 Support Vector Classifier [47]

Support Vector Classifier (SVC) is a type of supervised learning algorithm that belongs to the family of Support Vector Machines (SVMs). The goal of SVC is to find a hyperplane in a high-dimensional space that maximally separates the different classes of the data.

The hyperplane is determined by a subset of the training data points, called support vectors, which lie closest to the hyperplane. The support vectors define the optimal margin, which is the distance between the hyperplane and the closest points from each class. The ideal margin is selected so that it improves class separation while decreasing classification error.

SVC can handle both linearly separable and non-linearly separable datasets. For non-linearly separable datasets, SVC employs a method known as the kernel trick, which translates what is provided to a greater-dimensional environment where the classes may be separated by a linear hyperplane.

SVC is widely used in various applications such as image classification, text classification, and bioinformatics. One of the main advantages of SVC is its ability to handle high-dimensional data with a small number of samples. However, SVC can be sensitive to the choice of kernel function and its parameters, which may require some tuning.

3.3.10 Ada Boosting [48]

AdaBoost, is "Adaptive Boosting", is a ml method which blends numerous "weak" learner to build a "strong" learner. The technique works by training a series of weak ones repeatedly, with each new one emphasising the incorrectly classified cases from the prior stage. A graded mixture of these weak models yields the final "strong" classifier.

The basic idea behind AdaBoost is that, by combining many weak classifiers, the resulting "strong" classifier will be better than any of the individual weak classifiers. Each weak classifier is typically a simple decision tree or a linear model, and the weights assigned to each classifier are determined by their classification accuracy on the training data.

One of the advantages of AdaBoost is that it is a versatile algorithm that can be used with a wide variety of ml models. It has been shown to be particularly effective for tasks such as face detection, text classification, and speech recognition.

One of AdaBoost's drawbacks is that it is susceptible to noisy data as well as outliers. Furthermore, it can be computationally costly as each iteration requires training a new weak model on the full training set. Finally, it is important to note that AdaBoost is not a silver bullet, and there may be cases where other algorithms, such as Random Forests or Neural Networks, perform better.

3.3.11 Stochastic Gradient Descent [49]

Stochastic Gradient Descent (SGD) is a machine learning optimisation approach that is widely used to minimise the cost function associated with a model. It is a version of Gradient Descent, which is an iterative technique that updates the model parameters in the direction of the cost function's negative gradient in order to attain a local minimum.

The difference between Gradient Descent and Stochastic Gradient Descent is that, in SGD, the parameters are updated using a single randomly selected example (or a small batch of examples) from the training set instead of using the entire training set to compute the gradient. This makes SGD more computationally efficient and allows it to converge faster than Gradient Descent, especially for large datasets. However, SGD can be more noisy and can have more variance in the direction of the update since it only uses a subset of the data. To address this issue, a momentum term is often added to the update rule to smooth out the updates over time and make the convergence more stable.

Overall, SGD is a powerful optimization algorithm that is widely used in deep learning for training neural networks. It is efficient, scalable, and can handle large datasets with high-dimensional features.

3.4 Hyper Parameter Tunning [50]

Hyperparameter tuning is the process of selecting the optimal hyperparameters for a machine learning model. Hyperparameters are values set prior to training a model that affect the model's learning process and ultimately, its performance. Hyperparameter tuning involves searching for the optimal hyperparameters by testing different combinations of values and evaluating the model's performance on a validation set.

3.5 Performance Metrics [51,52]

Performance metrics play a crucial role in evaluating the effectiveness of Machine Learning (ML) models. They are used to assess the accuracy and reliability of a model's predictions. We have used some of performance metrics including accuracy, precision, recall, F1 score, ROC curve, and AUC. Accuracy measures the percentage of correctly classified instances in a dataset, while precision measures the percentage of true positives among all the predicted positives. Recall measures the percentage of true positives among all the actual positives, and the F1 score is the harmonic mean of precision and recall, providing a balanced measure of the two. ROC curve and AUC are used to evaluate the performance of classification models. The ROC curve plots the true positive rate against the false positive rate at different classification thresholds, while AUC measures the overall performance of the classifier, with higher AUC indicating better performance. By utilizing these performance metrics, ML practitioners can better evaluate and compare the performance of different models and select the most appropriate one for their specific use case.

3.6 Proposed Model

Ensemble learning is a powerful technique in machine learning that involves combining multiple models to achieve better predictive accuracy than any single model can achieve on its own [7, 8, 9]. It's a popular approach because it can reduce the risk of misclassifying instances and improve the overall accuracy of the classification process.

The idea behind ensemble learning is that multiple models can provide more accurate predictions when they work together. Just like how a sports team is stronger when it's composed of skilled players with different strengths and weaknesses, an ensemble of models can be more accurate because it combines the strengths of multiple models to reduce errors and biases.

There are several techniques for building ensembles, including stacking, boosting, voting, and bagging. We have used stacking in this project. The idea behind stacking is to use multiple base learners to generate predictions, and then use a meta learner to learn how to combine those predictions.

The process of building a stacking model involves several steps. First, we split the data into training and testing sets. Then, we train multiple base models on the training set and use them to generate predictions for the test set. We can then use those predictions as features to train the meta learner.

Finally, we test the stacking model on a holdout set. Stacking has obvious advantages. We can lower the risk of misclassification and enhance overall model accuracy by integrating the predictions of numerous models. This is particularly effective when the base models have different strengths, weaknesses, or assumptions about the data. The Algorithm of our proposed model is shown, table 2 shows the symbols used in algorithm 1. Figure 2 shows the proposed stack ensemble model.

Algorithm 1: Proposed Stack ensemble model

```
Input: W = {w1, w2, w3 ... wn} where w is the feature of the dataset
Output: 0 or 1                                          //non-diabetic or diabetic

Phase 1: Data pre-processing
X = removeoutliers(W)                      // removeoutliers() is to remove outliers
Y = removenull(X)                              // removenull() is to fill null values
Z = normalize(Y)                         // normalize() is to normalize the columns

Phase 2: Training the model
A = the 80% dataset for training, A ∈ Z                 //A is for training dataset
B = the 20% dataset for testing, B ∈ Z                   //B is for testing dataset

Models = {M1, M2, M3 ... Mn}               //the set of different Machine Learning
                                                models with hyper parameter tuning
for i = 1 to n do.
Begin.
Models(i).fit(A)               //training the Model on A with K fold cross validation
Next i
End

C = selectmodel()                                //selectmodel() is to select the top
                                                       5 ML models based on accuracy

for i = 1 to 5 do.
Begin
S = stacking(C, A)              //stacking() is to stack ensemble selected models
                                                              with training(A) dataset
Next i.
```

```
L = meta learner
$F = S \cup L$.                       //stacking S with the L (meta learner)
End.

Phase 3: Testing the model
F.predicts(B)                         //testing the stack ensemble model
                                      with B(testing dataset)
Result: F classifies B
```

Table 3: Algorithm 1 Symbol description

Serial Number	Symbols	Meaning
1	W	Dataset
2	X, Y, Z	Pre-processing variables
3	A	Training dataset
4	B	Testing dataset
5	M	Machine learning models
6	C	Selecting model Variable
7	S	Stacking
8	L	Meta learner

9	i	Iterator variable
10	F	Final model

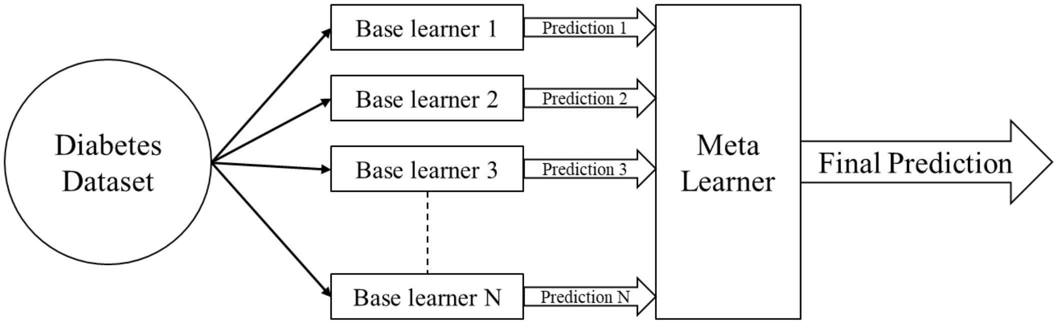


Figure 2: The proposed stack ensemble model for Diabetes prediction

CHAPTER-4 EXPERIMENTS & RESULT ANALYSIS

During this segment, we delved into a comprehensive assessment and analysis of the outcomes obtained from our proposed model. We made use of several performance metrics to gauge the efficiency of the algorithms utilized. Additionally, we contrasted the performance of our model with other established models, taking into account accuracy, precision, specificity, sensitivity, F1 Score, and AUC-ROC curve. It was also vital to examine the suggested model concerning other algorithms described in the literature review.

4.1 Model selection

Choosing the right model can be a challenging task, especially when you have to consider various factors such as data size, complexity, and performance. To make sure we chose the best model, we put a lot of effort into creating numerous baseline models which are gradient boosting classifier, decision tree classifier, multi-layer perceptron classifier, stochastic gradient descent, logistic regression, linear discriminant analysis, k-nearest neighbors, gaussian naive bayes, support vector machine, extra trees classifier, light gradient boosting machine, XGBoost, random forest classifier and Adaboost classifier, And evaluating their accuracies using cross-validation with 10-fold cross validation.

We finally picked the models with the highest accuracy rates, and the results are showcased in table 4. The process was not an easy one, but we believe that we made the best decision possible.

Table 4: Accuracy of models to be chosen for stacking

S. No.	Algorithm	Accuracy
1.	**XGB_2000**	**95.63%**
2.	**XGB_500**	**94.84%**
3.	**XGB_1000**	**95.23%**
4.	**DTC**	**95.50%**
5.	**LGBMC**	**95.52%**
6.	**RF_ENT100**	**95.90%**
7.	**RF_GINI100**	**95.90%**
8.	**ET100**	**96.03%**
9.	**ET500**	**96.03%**
10.	**ET1000**	**95.90%**
11.	LR_L2	89.95%
12.	LDA	88.63%
13.	KNN7	82.16%
14.	KNN5	81.90%
15.	KNN9	82.17%
16.	KNN11	80.97%
17.	LGBMC	95.50%
18.	SVM LINEAR	89.03%
19.	SVM RBF	90.88%
20.	AB	89.03%
21.	MLP	91.14%
22.	SGD3000	79.92%

To implement the stack ensemble method, we have carefully chosen different meta learner classifiers from our base learners, namely the Extra Tree Classifier, XGBoost, and Random Forest. These three classifiers have been optimized with distinct hyper parameter tunning than base learners to ensure a well-rounded ensemble. After thorough testing and evaluation, we have computed the accuracies of our stack ensemble method with each of these meta learner and have presented the detailed results in Table 5.

Table 5: Accuracy with different Meta learners

Level 1	Meta Learner	Accuracy
Random Forest Extra Tree Classifier Decision Tree XGBoost LightGBM	XGBoost	97.01%
	Extra Tree Classifier	**97.70%**
	Random Forest	96.58%

4.2 Results of Stack Ensemble model and it's comparison

The process of building a model can be quite challenging, but the satisfaction of seeing it produce accurate results can be quite rewarding. We built a model and were excited to find that it had an accuracy rate of 97.70%. This was a great achievement for us, but we knew that we needed to compare our model's performance with others to fully understand its effectiveness.

To do this, we used various performance measures like Precision, ROC-AUC, Sensitivity, Specificity, F1 Score, and Log Loss, which all provided valuable insights into the model's performance. For instance, our Precision score was 94.48%, indicating that the model was precise in its predictions. Similarly, our ROC-AUC score was 96.76%, which meant that the model's ability to distinguish between positive and negative classes was high.

We also found that our model's Sensitivity score was 93.48%, indicating that it was effective in identifying positive cases, and its Specificity score was 98.07%, meaning that it was good at identifying negative cases. finally, the F1 Score, which considers both precision and recall, was also impressive, at 93.45%.

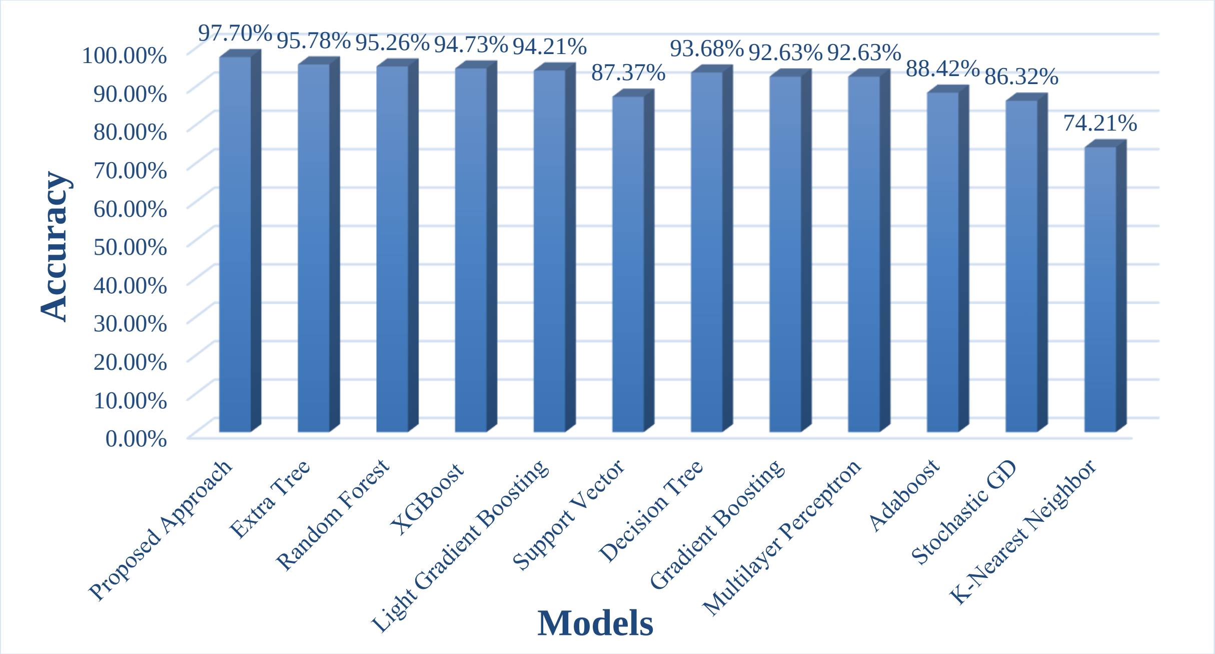
To gain a better understanding of our model's effectiveness, we compared its performance with other models that were used in the same context. We presented our results in Table 6, which showed the values of the various performance measures for each model.

Table 6: Machine learning algorithms and their accuracies compared with our model

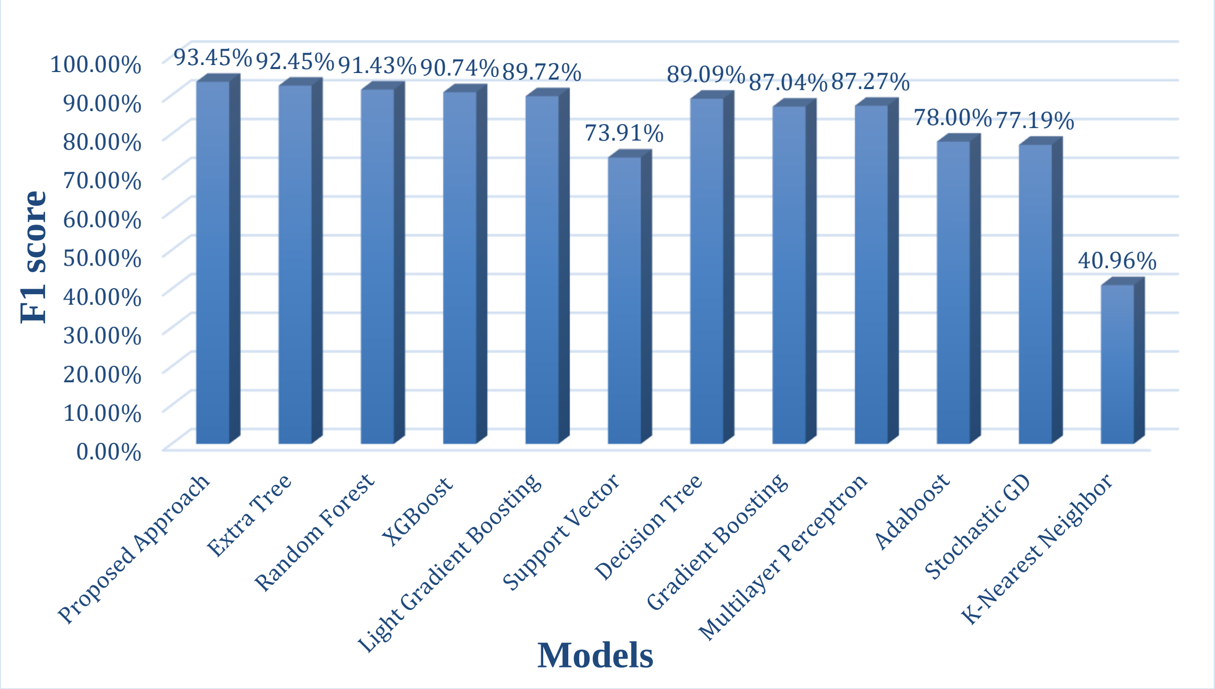
Model	Accuracy	Precision	ROC-AUC	Sensitivity	Specificity	F1 Score
Proposed Approach	97.70%	94.48%	96.76%	93.48%	98.07%	93.45%
Extra Tree	95.78%	92.45%	94.77%	92.45%	97.08%	92.45%
Random Forest	95.26%	92.31%	93.82%	90.57%	97.08%	91.43%
XGBoost	94.73%	89.09%	94.04%	92.45%	95.62%	90.74%
Light Gradient Boosting	94.21%	88.89%	93.09%	90.57%	95.62%	89.72%

Support Vector	87.37%	87.18%	80.25%	64.15%	96.35%	73.91%
Decision Tree	93.68%	85.96%	93.31%	92.45%	94.16%	89.09%
Gradient Boosting	92.63%	85.45%	91.42%	88.68%	94.16%	87.04%
Multilayer Perceptron	92.63%	84.21%	92.00%	90.57%	93.43%	87.27%
Adaboost	88.42%	82.98%	83.87%	73.58%	94.16%	78.00%
Stochastic GD	86.32%	72.13%	85.31%	83.02%	87.59%	77.19%
K-Nearest Neighbor	74.21%	56.67%	61.29%	32.08%	90.51%	40.96%

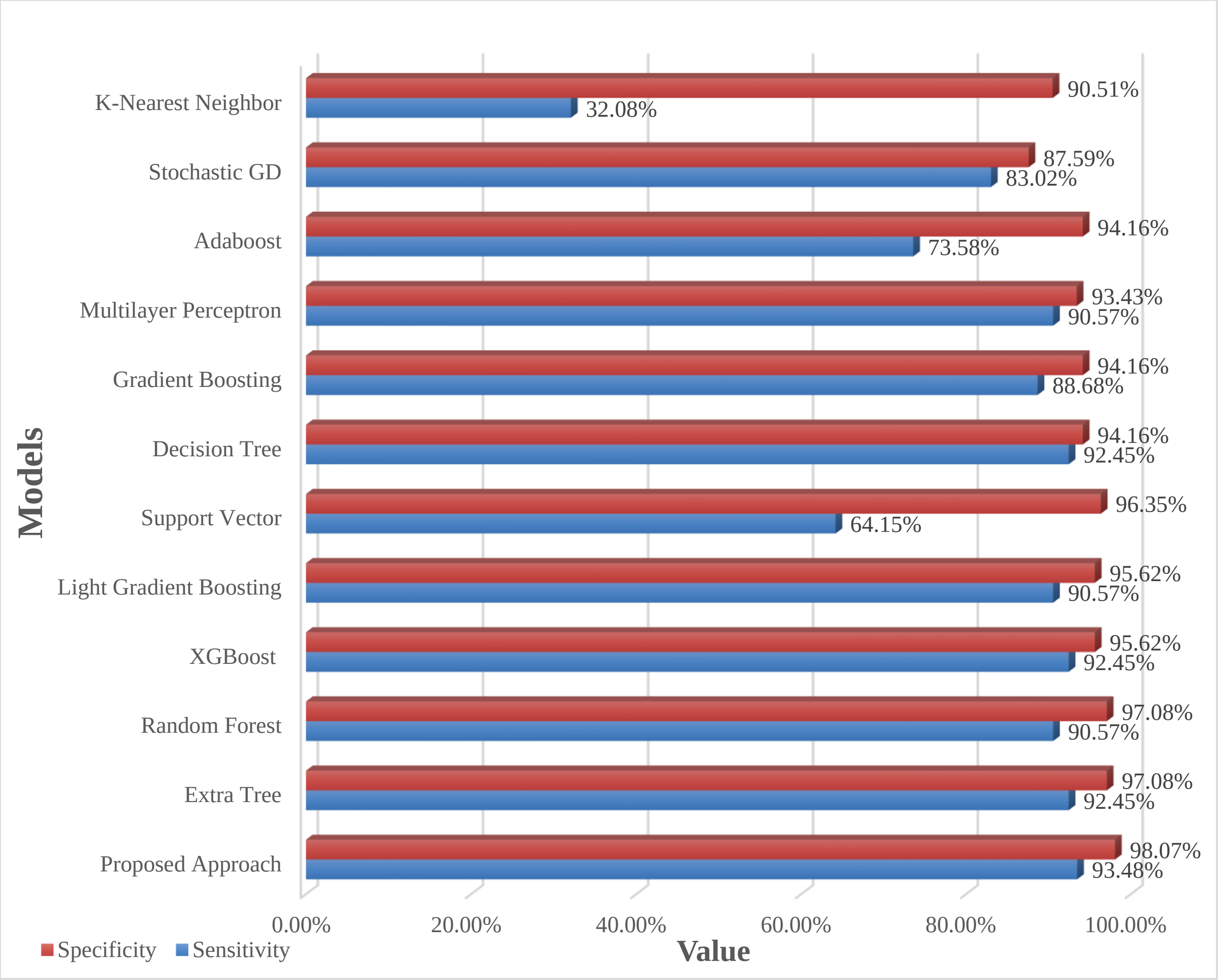
When it comes to analyzing data models, visualization tools like graphs can be quite handy. In this case, we've plotted the accuracies of all the models, including the stack ensemble model, in graph 10 and F1 score in graph 11. But, that's not all, to delve deeper into the insights, we've also created graph 12 where we've plotted sensitivity & specificity. This graph shows us how the models performed in correctly identifying positive and negative outcomes. Similarly, in graph 13, we've plotted ROC-AUC & Precision, which gives us a better understanding of how well the models perform in classifying true positives and true negatives.



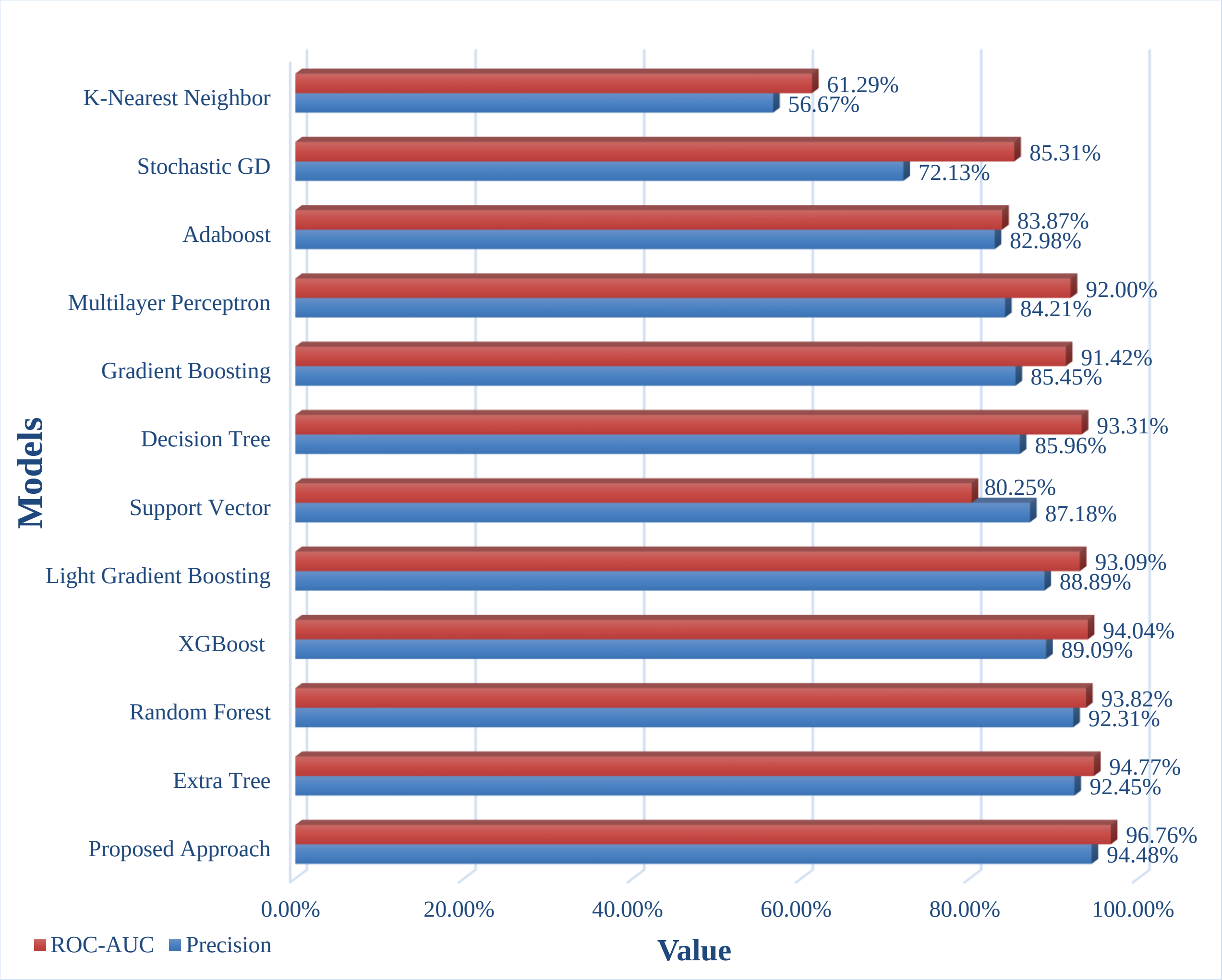
Graph 10: Comparison of accuracy of proposed stack ensemble model with existing ML models



Graph 11: Comparison of F1 Score of proposed stack ensemble model with existing ML models

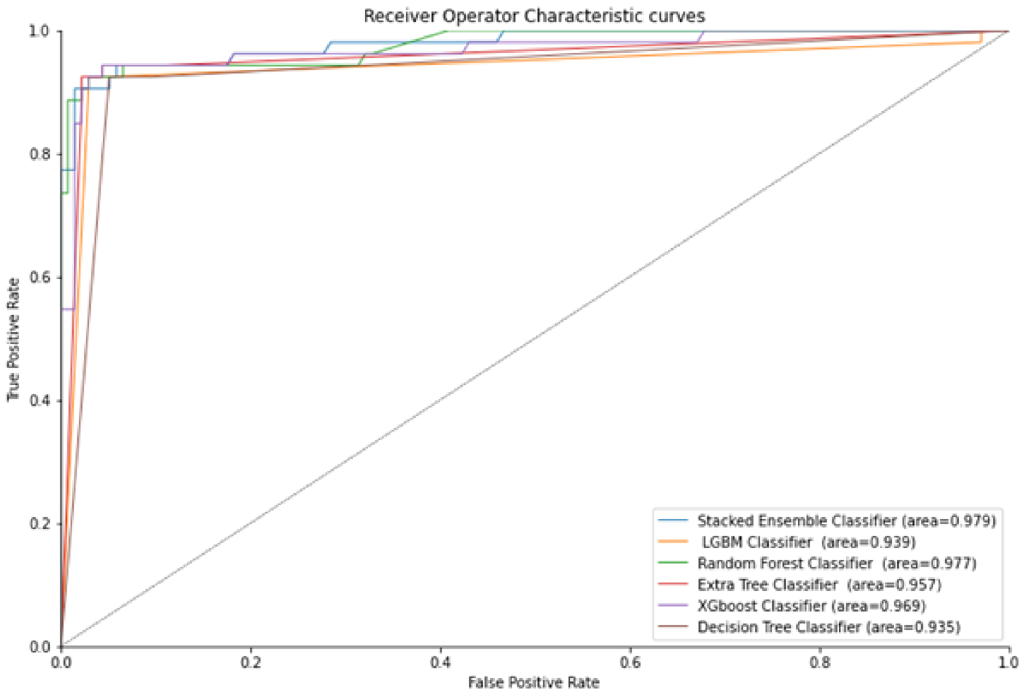


Graph 12: Comparison of sensitivity and specificity of proposed stack ensemble model with different ML models

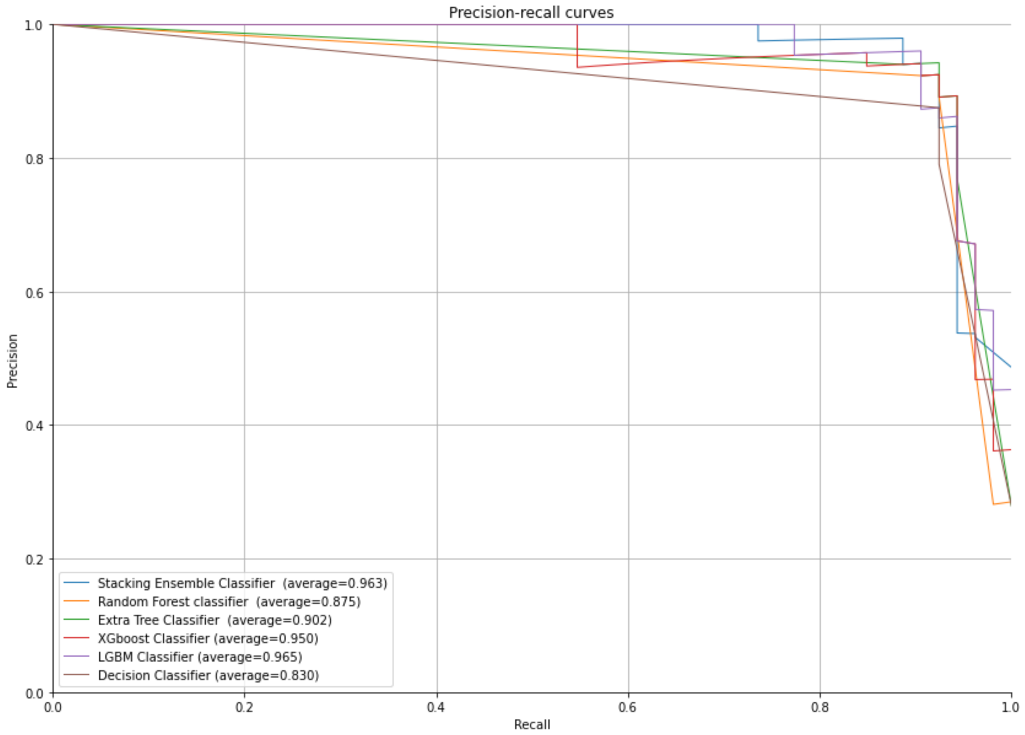


Graph 13: Comparison of ROC-AUC and Precision of proposed stack ensemble model with different ML models

The AUC-ROC curves and Precision-Recall curve for the stack ensemble model are represented in graphs 14 and 15. These graphs offer a visual representation of how the stack ensemble model performed in comparison to other top-performing models.

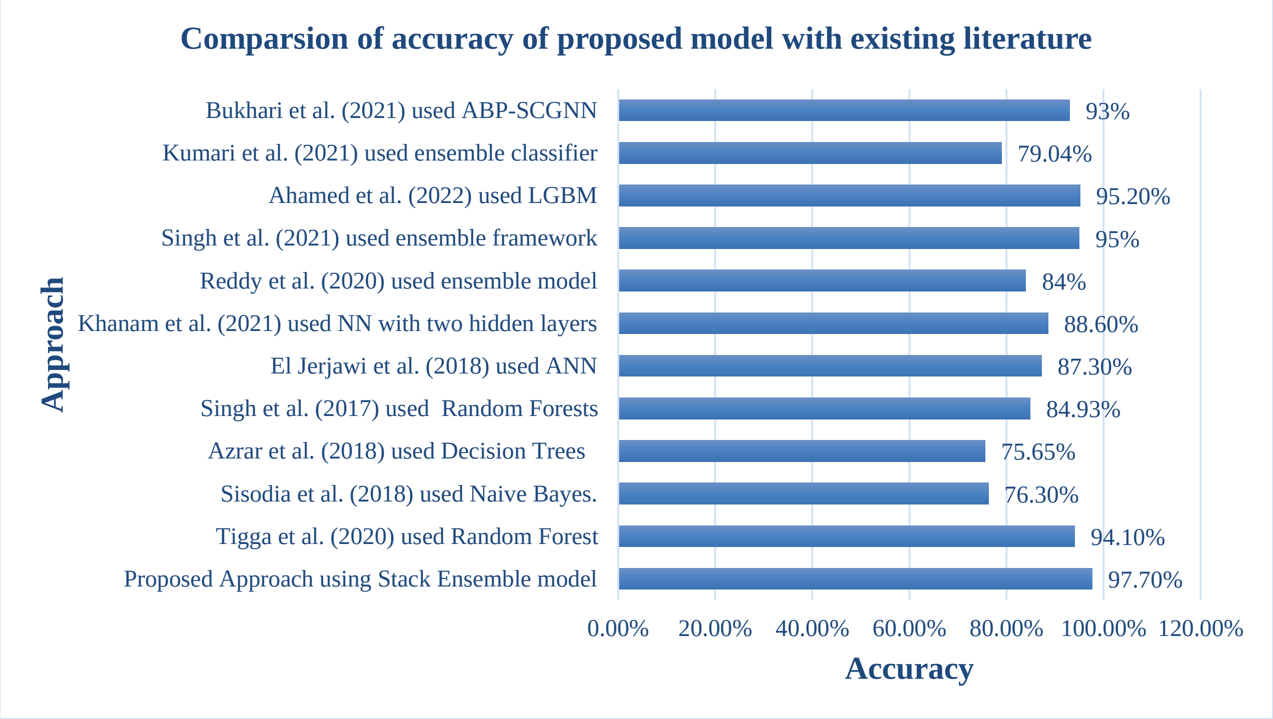


Graph 14: AUC-ROC curves of stack ensemble model and top performing models



Graph 15: Precision-Recall curve of stack ensemble model and top performing models

After spending countless hours spending over research papers and fine-tuning our algorithm, we were finally ready to compare our results with those of our peers. We presented our findings in Graph 16 and juxtaposed them against the approaches covered in Table 1. And just as we had hoped, Graph 16 clearly showed that our model was the best performer among all the approaches we had studied.



Graph 16: Comparison of results of our approach with others approaches covered in Table 1.

CHAPTER-5 CONCLUSIONS

5.1 Conclusions

Diabetes prediction at an early stage is very important as it can lead to various health conditions if left undetected. The proposed framework is used to predict whether a person is diabetic or not taking various features into account. The dataset is picked from Kaggle, diabetes 2019 dataset, and data pre-processing done first. Then various classification algorithms are applied and their accuracy is compared. Extra forest classifiers gave the best accuracy of 95.78% among them. Later an stack ensemble model is proposed and is built using extra tree classifier, random forest, decision tree, XGBoost, LightGBM as base learners and extra tree classifier as the meta learner. The proposed model outperformed all the existing models and gave an accuracy of 97.70%. The proposed model also performed better than the existing state-of-art literature. Precision, Sensitivity, Specificity, AUC-Roc curve, F1 score are also taken into account to compare the proposed model with already existing models and our model performed best among the existing ones.

5.2 Future Scope

- The model can be improved by collecting information from nearby hospitals and building our own dataset which would result in a more optimised model.
- At present, we have utilized two-class stacking, but we can enhance the model's performance by using multiclass stacking. Furthermore, we can enhance accuracy by working with datasets of varying dimensions and incorporating diverse models with distinct hyper parameters.
- Along with k fold validation technique while implementing ensembling, blending technique can also be applied which might lead to more improved results.

- After achieving significant accuracy, a website can be built for early prediction of diabetes. This would involve a person putting different input like his age, amount of physical activity, regular medicine, if he smokes or drink and other features. The trained model would therefore predict whether a person is diabetic or not.

5.3 Applications Contributions

We proposed a Stacking based Ensemble Learning model which increases classifier diversity. A stacking approach is employed to determine the optimal way of combining the predictions of multiple machine learning algorithms. The base learner utilized include Random Forest, XGBoost, LightGBM, decision tree, and extra tree classifier, which are subsequently fed into the meta learner. The resulting model gave an accuracy of 97.70% which outperformed all the existing conventional models and even ensemble models.

We have used a better dataset than the mostly used i.e., PIMA dataset as our dataset is more reliable, contains more attributes, and is larger.

Also, to determine the optimal parameter for machine learning model training, a technique of Hyper Parameter Tuning is implemented.

The stacked ensemble model is compared with existing ML models on the basis of AUC-ROC, Precision, F1 score, precision, Specificity sensitivity, MCC & Accuracy and also compared with the literature covered.

REFERENCES

1. De Silva, Dilani D., et al. "Medicinal mushrooms in prevention and control of diabetes mellitus." *Fungal Diversity* 56 (2012): 1-29.
2. Alam, Talha Mahboob, et al. "A model for early prediction of diabetes." *Informatics in Medicine Unlocked* 16 (2019): 100204.
3. Ranasinghe, Priyanga, et al. "Prevalence and trends of the diabetes epidemic in urban and rural India: A pooled systematic review and meta-analysis of 1.7 million adults." *Annals of epidemiology* 58 (2021): 128-148.
4. Akil, A.AS., Yassin, E., Al-Maraghi, A. et al. Diagnosis and treatment of type 1 diabetes at the dawn of the personalized medicine era. J Transl Med 19, 137 (2021). https://doi.org/10.1186/s12967-021-02778-6
5. Olokoba AB, Obateru OA, Olokoba LB. Type 2 diabetes mellitus: a review of current trends. Oman Med J. 2012 Jul;27(4):269-73. doi: 10.5001/omj.2012.68.
6. McIntyre, H. D., Kapur, A., Divakar, H., & Hod, M. (1AD, January 1). Gestational diabetes mellitus-innovative approach to prediction, diagnosis, management, and prevention of future NCD-mother and offspring. Frontiers. Retrieved from https://www.frontiersin.org/articles/10.3389/fendo.2020.614533/full
7. Abdollahi, Jafar, and Babak Nouri-Moghaddam. "Hybrid stacked ensemble combined with genetic algorithms for diabetes prediction." *Iran Journal of Computer Science* 5.3 (2022): 205-220.
8. Tiwari, Achyut, Aryan Chugh, and Aman Sharma. "Ensemble framework for cardiovascular disease prediction." *Computers in Biology and Medicine* 146 (2022): 105624.
9. VK, Daliya, and T. K. Ramesh. "Optimized stacking ensemble models for the prediction of diabetic progression." *Multimedia Tools and Applications* (2023): 1-25.

10. Tigga, Neha Prerna, and Shruti Garg. "Prediction of type 2 diabetes using machine learning classification methods." *Procedia Computer Science* 167 (2020): 706-716.
11. Sisodia, Deepti, and Dilip Singh Sisodia. "Prediction of diabetes using classification algorithms." *Procedia computer science* 132 (2018): 1578-1585.
12. Yang, Hui, et al. "Risk prediction of diabetes: big data mining with fusion of multifarious physical examination indicators." *Information Fusion* 75 (2021): 140-149.
13. Azrar, Amina, et al. "Data mining models comparison for diabetes prediction." *International Journal of Advanced Computer Science and Applications* 9.8 (2018).
14. Singh, D. A. A. G., E. Jebamalar Leavline, and B. Shanawaz Baig. "Diabetes prediction using medical data." *Journal of Computational Intelligence in Bioinformatics* 10.1 (2017): 1-8.
15. Ayon, Safial Islam, and Md Milon Islam. "Diabetes prediction: a deep learning approach." *International Journal of Information Engineering and Electronic Business* 12.2 (2019): 21.
16. El_Jerjawi, Nesreen Samer, and Samy S. Abu-Naser. "Diabetes prediction using artificial neural network." (2018).
17. Khanam, Jobeda Jamal, and Simon Y. Foo. "A comparison of machine learning algorithms for diabetes prediction." *ICT Express* 7.4 (2021): 432-439.
18. Reddy, G. Thippa, et al. "An ensemble based machine learning model for diabetic retinopathy classification." *2020 international conference on emerging trends in information technology and engineering (ic-ETITE)*. IEEE, 2020.
19. Sarwar, Abid, et al. "Diagnosis of diabetes type-II using hybrid machine learning based ensemble model." *International Journal of Information Technology* 12 (2020): 419-428.

20. Singh, Ashima, et al. "eDiaPredict: an ensemble-based framework for diabetes prediction." *ACM Transactions on Multimidia Computing Communications and Applications* 17.2s (2021): 1-26.
21. Ahamed, B. Shamreen, Meenakshi Sumeet Arya, and Auxilia Osvin Nancy V. "Prediction of Type-2 Diabetes Mellitus Disease Using Machine Learning Classifiers and Techniques." *Frontiers in Computer Science* 4 (2022).
22. Kumari, Saloni, Deepika Kumar, and Mamta Mittal. "An ensemble approach for classification and prediction of diabetes mellitus using soft voting classifier." *International Journal of Cognitive Computing in Engineering* 2 (2021): 40-46.
23. Bukhari, Muhammad Mazhar, et al. "An improved artificial neural network model for effective diabetes prediction." *Complexity* 2021 (2021): 1-10.
24. Hasan, Md Kamrul, et al. "Diabetes prediction using ensembling of different machine learning classifiers." *IEEE Access* 8 (2020): 76516-76531.
25. Nai-Arun, Nongyao, and Punnee Sittidech. "Ensemble learning model for diabetes classification." *Advanced Materials Research*. Vol. 931. Trans Tech Publications Ltd, 2014.
26. Sisodia, Deepti, and Dilip Singh Sisodia. "Prediction of diabetes using classification algorithms." *Procedia computer science* 132 (2018): 1578-1585.
27. Chen, Xin-Xin, et al. "Identification of bacterial cell wall lyases via pseudo amino acid composition." *BioMed research international* 2016 (2016).
28. Habibi, Shafi, Maryam Ahmadi, and Somayeh Alizadeh. "Type 2 diabetes mellitus screening and risk factors using decision tree: results of data mining." *Global journal of health science* 7.5 (2015): 304.
29. Sivaranjani, S., et al. "Diabetes prediction using machine learning algorithms with feature selection and dimensionality reduction." *2021 7th*

International Conference on Advanced Computing and Communication Systems (ICACCS). Vol. 1. IEEE, 2021.

30. Gollapalli, Mohammed, et al. "A novel stacking ensemble for detecting three types of diabetes mellitus using a Saudi Arabian dataset: pre-diabetes, T1DM, and T2DM." *Computers in Biology and Medicine* 147 (2022): 105757.
31. Rajendra, Priyanka, and Shahram Latifi. "Prediction of diabetes using logistic regression and ensemble techniques." *Computer Methods and Programs in Biomedicine Update* 1 (2021): 100032.
32. Husain, Adil, and Muneeb H. Khan. "Early diabetes prediction using voting based ensemble learning." *Advances in Computing and Data Sciences: Second International Conference, ICACDS 2018, Dehradun, India, April 20-21, 2018, Revised Selected Papers, Part I 2*. Springer Singapore, 2018.
33. Mahesh, T. R., et al. "Blended ensemble learning prediction model for strengthening diagnosis and treatment of chronic diabetes disease." *Computational Intelligence and Neuroscience* 2022 (2022).
34. Ali, Rahman, et al. "Prediction of diabetes mellitus based on boosting ensemble modeling." *Ubiquitous Computing and Ambient Intelligence. Personalisation and User Adapted Services: 8th International Conference, UCAmI 2014, Belfast, UK, December 2-5, 2014. Proceedings 8*. Springer International Publishing, 2014.
35. Dutta, Aishwariya, et al. "Early prediction of diabetes using an ensemble of machine learning models." *International Journal of Environmental Research and Public Health* 19.19 (2022): 12378.
36. Saxena, Roshi, et al. "A novel approach for feature selection and classification of diabetes mellitus: Machine learning methods." *Computational Intelligence and Neuroscience* 2022 (2022).
37. Maniruzzaman, Md, et al. "Classification and prediction of diabetes disease using machine learning paradigm." *Health information science and systems* 8 (2020): 1-14.

38. Laila, Umm E., et al. "An ensemble approach to predict early-stage diabetes risk using machine learning: An empirical study." *Sensors* 22.14 (2022): 5247.
39. Rogers, Jeremy, and Steve Gunn. "Identifying feature relevance using a random forest." *Subspace, Latent Structure and Feature Selection: Statistical and Optimization Perspectives Workshop, SLSFS 2005, Bohinj, Slovenia, February 23-25, 2005, Revised Selected Papers*. Springer Berlin Heidelberg, 2006.
40. Bhati, Bhoopesh Singh, and C. S. Rai. "Ensemble based approach for intrusion detection using extra tree classifier." *Intelligent Computing in Engineering: Select Proceedings of RICE 2019*. Springer Singapore, 2020.
41. Safavian, S. Rasoul, and David Landgrebe. "A survey of decision tree classifier methodology." *IEEE transactions on systems, man, and cybernetics* 21.3 (1991): 660-674.
42. Ahamed, B. Shamreen. "Prediction of type-2 diabetes using the LGBM classifier methods and techniques." *Turkish Journal of Computer and Mathematics Education (TURCOMAT)* 12.12 (2021): 223-231.
43. Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016.
44. Friedman, Jerome H. "Greedy function approximation: a gradient boosting machine." *Annals of statistics* (2001): 1189-1232.
45. Zhang, Min-Ling, and Zhi-Hua Zhou. "A k-nearest neighbor based algorithm for multi-label classification." *2005 IEEE international conference on granular computing*. Vol. 2. IEEE, 2005.
46. Amendolia, Salvator Roberto, et al. "A comparative study of k-nearest neighbour, support vector machine and multi-layer perceptron for thalassemia screening." *Chemometrics and Intelligent Laboratory Systems* 69.1-2 (2003): 13-20.

47. Vijayakumar, Sethu, and Si Wu. "Sequential Support Vector Classifiers and Regression." *IIA/SOCO*. 1999.

48. Rätsch, Gunnar, Takashi Onoda, and K-R. Müller. "Soft margins for AdaBoost." *Machine learning* 42 (2001): 287-320.

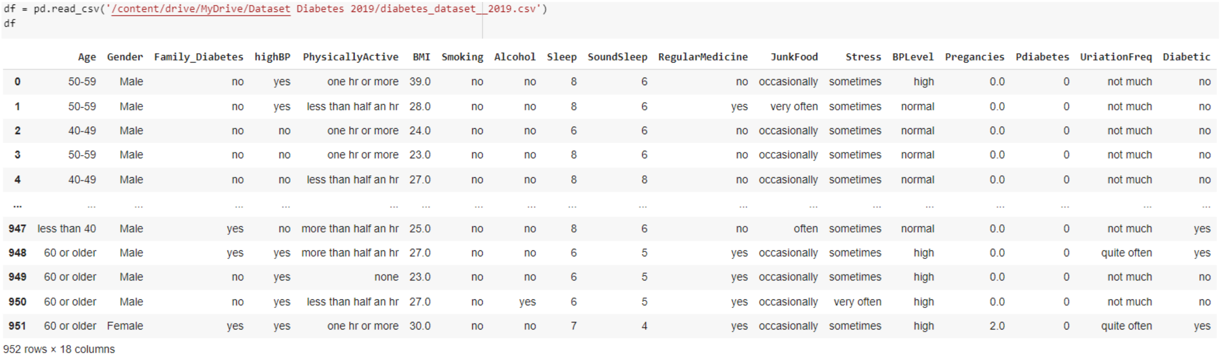
49. Dorion, Christian, and Yoshua Bengio. "Stochastic Gradient Descent on a Portfolio Management Training Criterion Using the IPA Gradient Estimator." (2003).

50. Friedrichs, Frauke, and Christian Igel. "Evolutionary tuning of multiple SVM parameters." *Neurocomputing* 64 (2005): 107-117.

51. Ramakrishnan, Navya. "An intelligent system for early detection of eye diseases that lead to irreversible vision loss." (2020).

52. Sisodia, Deepti, and Dilip Singh Sisodia. "Prediction of diabetes using classification algorithms." *Procedia computer science* 132 (2018): 1578-1585.

APPENDICES

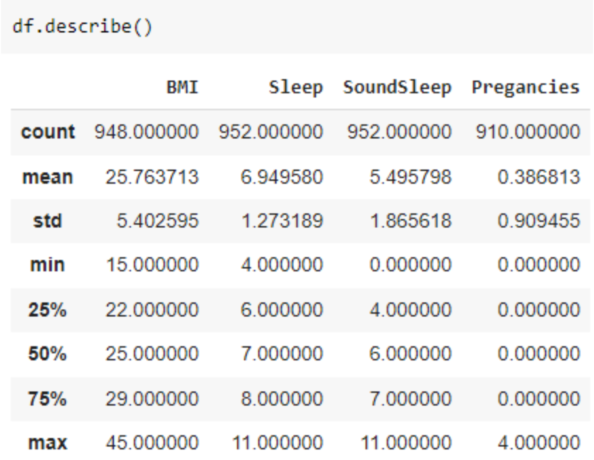


```
df = pd.read_csv('/content/drive/MyDrive/Dataset Diabetes 2019/diabetes_dataset__2019.csv')
df
```

	Age	Gender	Family_Diabetes	highBP	PhysicallyActive	BMI	Smoking	Alcohol	Sleep	SoundSleep	RegularMedicine	JunkFood	Stress	BPLevel	Pregancies	Pdiabetes	UriationFreq	Diabetic
0	50-59	Male	no	yes	one hr or more	39.0	no	no	8	6	no	occasionally	sometimes	high	0.0	0	not much	no
1	50-59	Male	no	yes	less than half an hr	28.0	no	no	8	6	yes	very often	sometimes	normal	0.0	0	not much	no
2	40-49	Male	no	no	one hr or more	24.0	no	no	6	6	no	occasionally	sometimes	normal	0.0	0	not much	no
3	50-59	Male	no	no	one hr or more	23.0	no	no	8	6	no	occasionally	sometimes	normal	0.0	0	not much	no
4	40-49	Male	no	no	less than half an hr	27.0	no	no	8	8	no	occasionally	sometimes	normal	0.0	0	not much	no
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
947	less than 40	Male	yes	no	more than half an hr	25.0	no	no	8	6	no	often	sometimes	normal	0.0	0	not much	yes
948	60 or older	Male	yes	yes	more than half an hr	27.0	no	no	6	5	yes	occasionally	sometimes	high	0.0	0	quite often	yes
949	60 or older	Male	no	yes	none	23.0	no	no	6	5	yes	occasionally	sometimes	high	0.0	0	not much	no
950	60 or older	Male	no	yes	less than half an hr	27.0	no	yes	6	5	yes	occasionally	very often	high	0.0	0	not much	no
951	60 or older	Female	yes	yes	one hr or more	30.0	no	no	7	4	yes	occasionally	sometimes	high	2.0	0	quite often	yes

952 rows × 18 columns

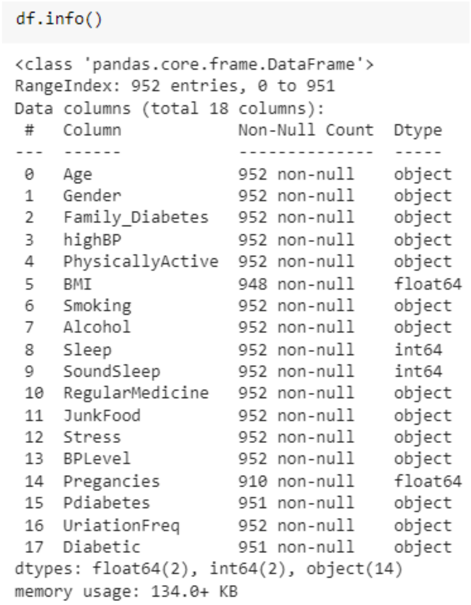
Code figure 1



```
df.describe()
```

	BMI	Sleep	SoundSleep	Pregancies
count	948.000000	952.000000	952.000000	910.000000
mean	25.763713	6.949580	5.495798	0.386813
std	5.402595	1.273189	1.865618	0.909455
min	15.000000	4.000000	0.000000	0.000000
25%	22.000000	6.000000	4.000000	0.000000
50%	25.000000	7.000000	6.000000	0.000000
75%	29.000000	8.000000	7.000000	0.000000
max	45.000000	11.000000	11.000000	4.000000

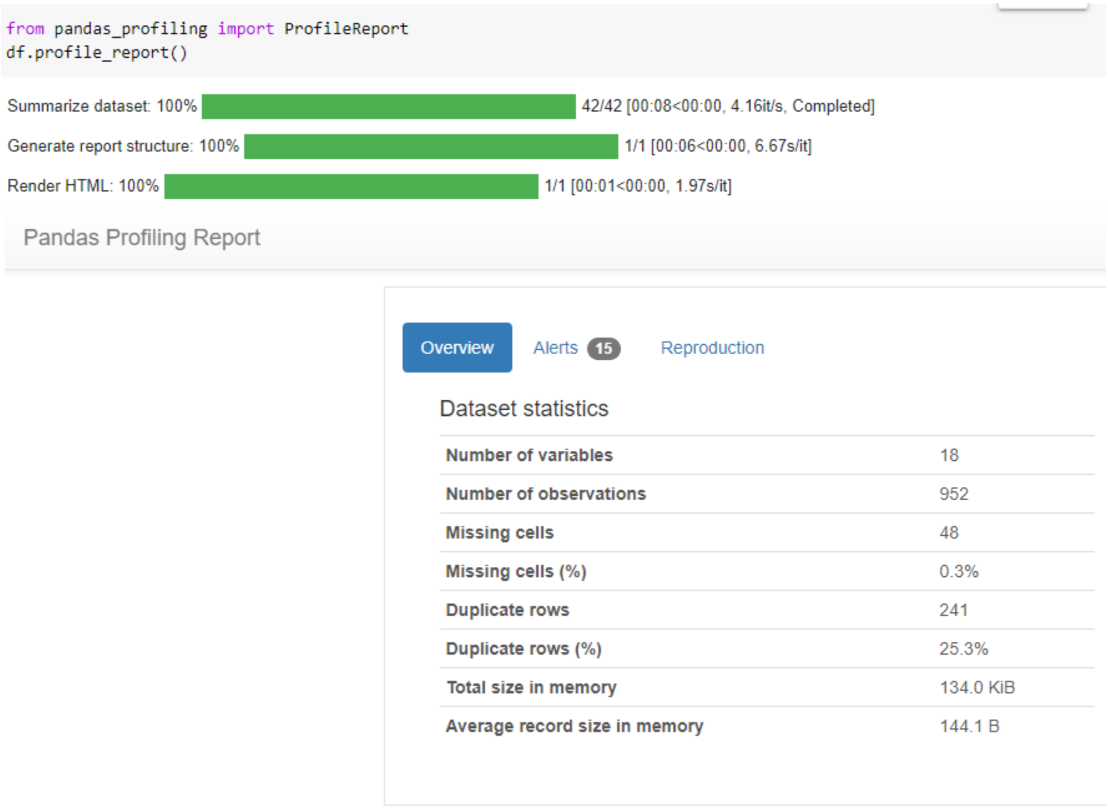
Code figure 2



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 952 entries, 0 to 951
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Age               952 non-null    object 
 1   Gender            952 non-null    object 
 2   Family_Diabetes   952 non-null    object 
 3   highBP            952 non-null    object 
 4   PhysicallyActive  952 non-null    object 
 5   BMI               948 non-null    float64
 6   Smoking           952 non-null    object 
 7   Alcohol           952 non-null    object 
 8   Sleep             952 non-null    int64  
 9   SoundSleep        952 non-null    int64  
 10  RegularMedicine   952 non-null    object 
 11  JunkFood          952 non-null    object 
 12  Stress            952 non-null    object 
 13  BPLevel           952 non-null    object 
 14  Pregancies        910 non-null    float64
 15  Pdiabetes         951 non-null    object 
 16  UriationFreq      952 non-null    object 
 17  Diabetic          951 non-null    object 
dtypes: float64(2), int64(2), object(14)
memory usage: 134.0+ KB
```

Code figure 3



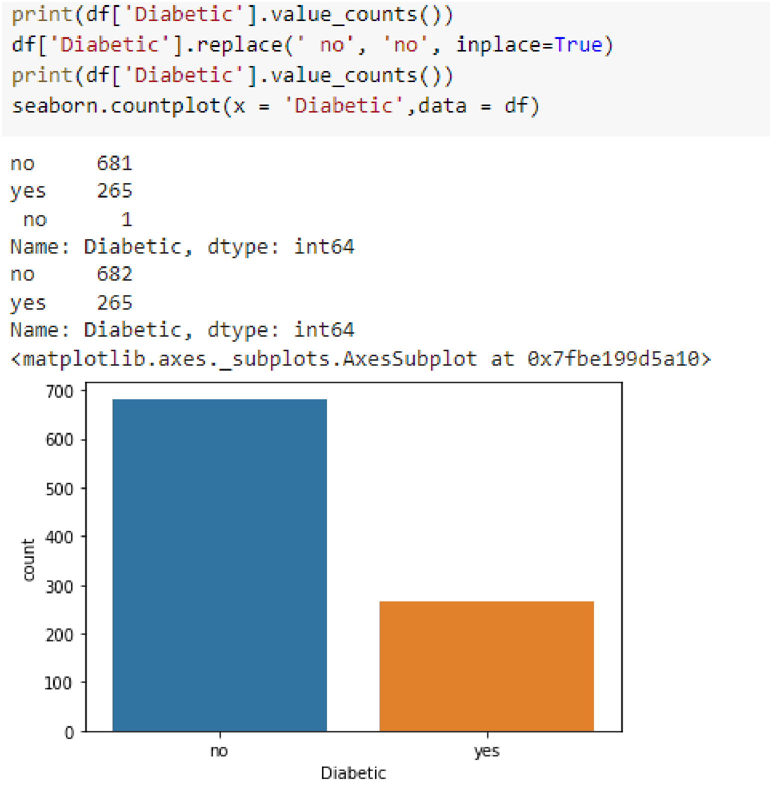
```
from pandas_profiling import ProfileReport
df.profile_report()
```

```
Summarize dataset: 100%  42/42 [00:08<00:00, 4.16it/s, Completed]
Generate report structure: 100%  1/1 [00:06<00:00, 6.67s/it]
Render HTML: 100%  1/1 [00:01<00:00, 1.97s/it]
```

Dataset statistics

Number of variables	18
Number of observations	952
Missing cells	48
Missing cells (%)	0.3%
Duplicate rows	241
Duplicate rows (%)	25.3%
Total size in memory	134.0 KiB
Average record size in memory	144.1 B

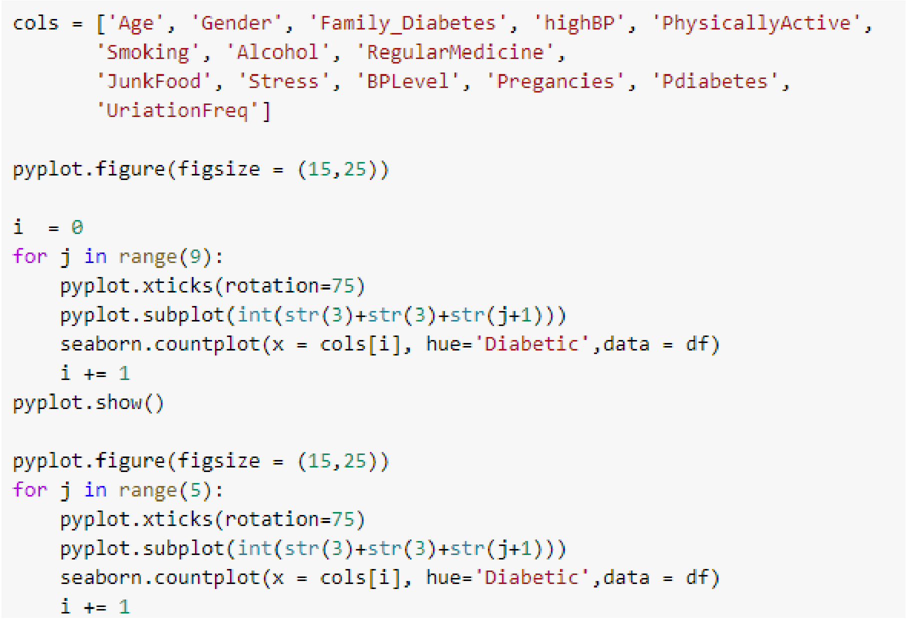
Code figure 4



```python
print(df['Diabetic'].value_counts())
df['Diabetic'].replace(' no', 'no', inplace=True)
print(df['Diabetic'].value_counts())
seaborn.countplot(x = 'Diabetic',data = df)
```

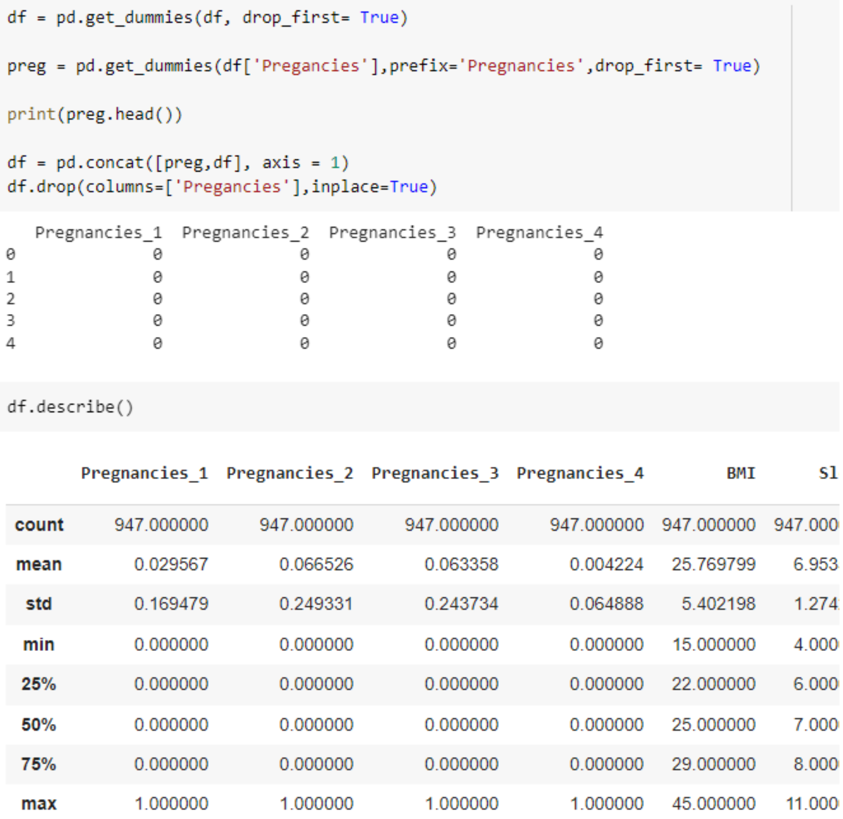
```
no      681
yes     265
 no       1
Name: Diabetic, dtype: int64
no      682
yes     265
Name: Diabetic, dtype: int64
<matplotlib.axes._subplots.AxesSubplot at 0x7fbe199d5a10>
```

Code figure 5



```python
cols = ['Age', 'Gender', 'Family_Diabetes', 'highBP', 'PhysicallyActive',
       'Smoking', 'Alcohol', 'RegularMedicine',
       'JunkFood', 'Stress', 'BPLevel', 'Pregancies', 'Pdiabetes',
       'UriationFreq']

pyplot.figure(figsize = (15,25))

i  = 0
for j in range(9):
    pyplot.xticks(rotation=75)
    pyplot.subplot(int(str(3)+str(3)+str(j+1)))
    seaborn.countplot(x = cols[i], hue='Diabetic',data = df)
    i += 1
pyplot.show()

pyplot.figure(figsize = (15,25))
for j in range(5):
    pyplot.xticks(rotation=75)
    pyplot.subplot(int(str(3)+str(3)+str(j+1)))
    seaborn.countplot(x = cols[i], hue='Diabetic',data = df)
    i += 1
```

Code figure 6



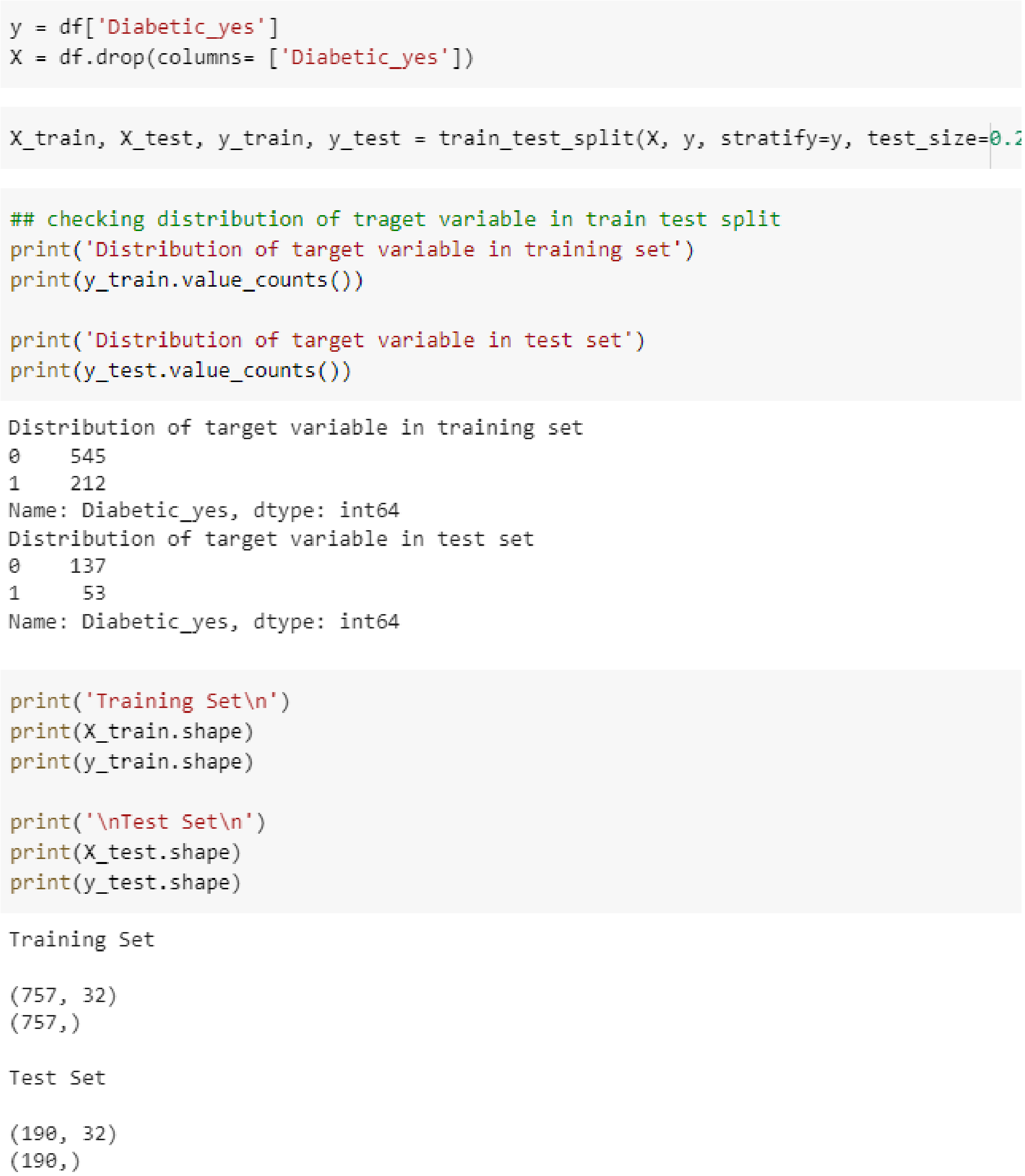
```
df = pd.get_dummies(df, drop_first= True)

preg = pd.get_dummies(df['Pregancies'],prefix='Pregnancies',drop_first= True)

print(preg.head())

df = pd.concat([preg,df], axis = 1)
df.drop(columns=['Pregancies'],inplace=True)
```

```
   Pregnancies_1  Pregnancies_2  Pregnancies_3  Pregnancies_4
0              0              0              0              0
1              0              0              0              0
2              0              0              0              0
3              0              0              0              0
4              0              0              0              0
```

```
df.describe()
```

	Pregnancies_1	Pregnancies_2	Pregnancies_3	Pregnancies_4	BMI	Sl
count	947.000000	947.000000	947.000000	947.000000	947.000000	947.000
mean	0.029567	0.066526	0.063358	0.004224	25.769799	6.953
std	0.169479	0.249331	0.243734	0.064888	5.402198	1.274
min	0.000000	0.000000	0.000000	0.000000	15.000000	4.000
25%	0.000000	0.000000	0.000000	0.000000	22.000000	6.000
50%	0.000000	0.000000	0.000000	0.000000	25.000000	7.000
75%	0.000000	0.000000	0.000000	0.000000	29.000000	8.000
max	1.000000	1.000000	1.000000	1.000000	45.000000	11.000

Code figure 7



```
y = df['Diabetic_yes']
X = df.drop(columns= ['Diabetic_yes'])
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.2
```

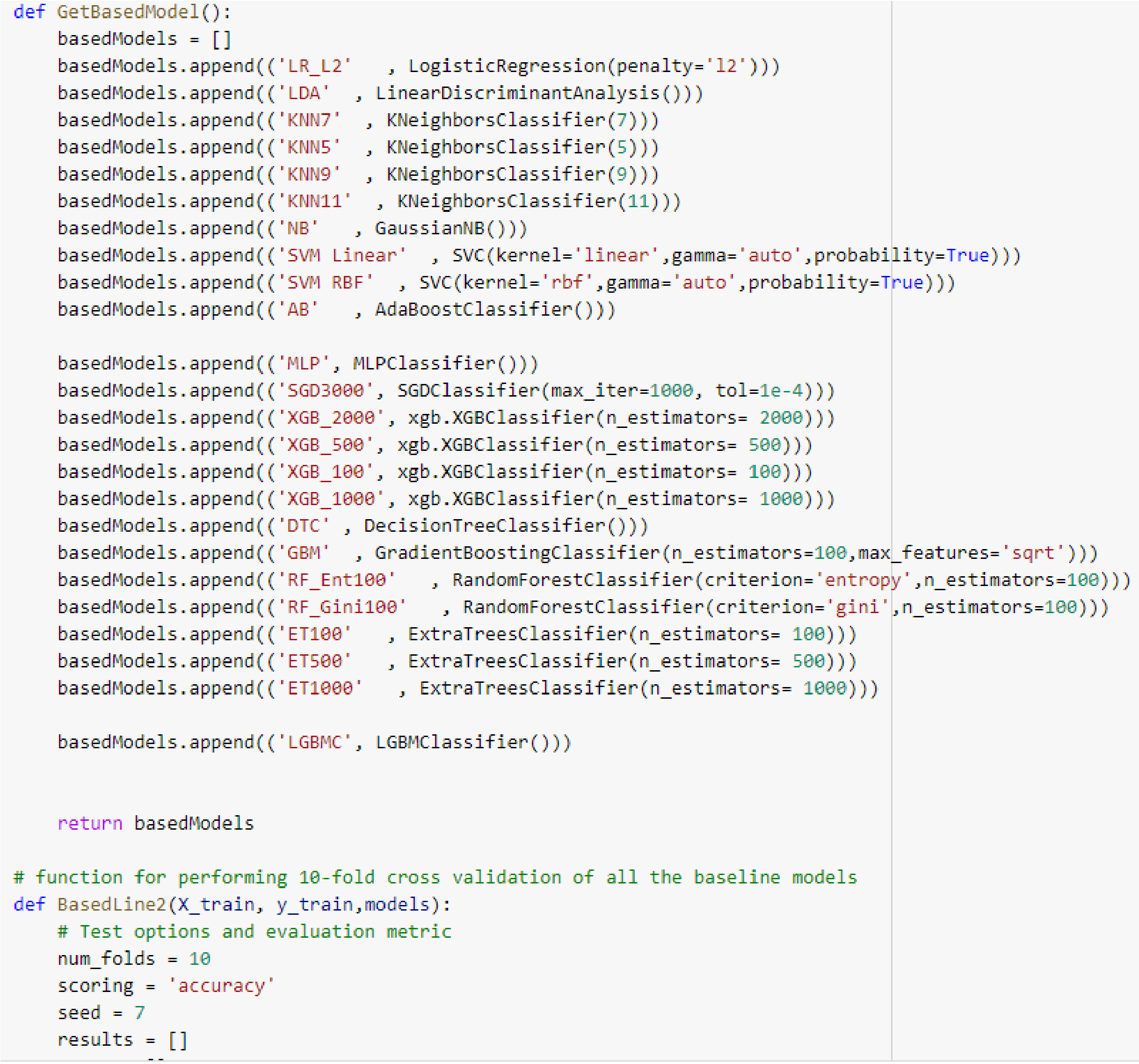
```
## checking distribution of traget variable in train test split
print('Distribution of target variable in training set')
print(y_train.value_counts())

print('Distribution of target variable in test set')
print(y_test.value_counts())
```

```
Distribution of target variable in training set
0    545
1    212
Name: Diabetic_yes, dtype: int64
Distribution of target variable in test set
0    137
1     53
Name: Diabetic_yes, dtype: int64
```

```
print('Training Set\n')
print(X_train.shape)
print(y_train.shape)

print('\nTest Set\n')
print(X_test.shape)
print(y_test.shape)
```

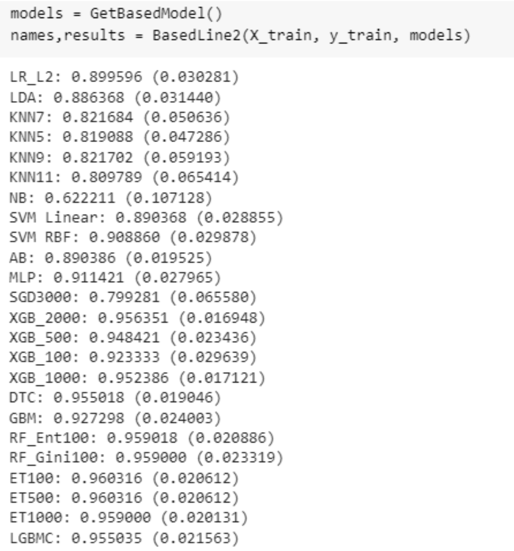
```
Training Set

(757, 32)
(757,)

Test Set

(190, 32)
(190,)
```

Code figure 8



```
def GetBasedModel():
    basedModels = []
    basedModels.append(('LR_L2'   , LogisticRegression(penalty='l2')))
    basedModels.append(('LDA'  , LinearDiscriminantAnalysis()))
    basedModels.append(('KNN7'  , KNeighborsClassifier(7)))
    basedModels.append(('KNN5'  , KNeighborsClassifier(5)))
    basedModels.append(('KNN9'  , KNeighborsClassifier(9)))
    basedModels.append(('KNN11'  , KNeighborsClassifier(11)))
    basedModels.append(('NB'   , GaussianNB()))
    basedModels.append(('SVM Linear'  , SVC(kernel='linear',gamma='auto',probability=True)))
    basedModels.append(('SVM RBF'  , SVC(kernel='rbf',gamma='auto',probability=True)))
    basedModels.append(('AB'   , AdaBoostClassifier()))

    basedModels.append(('MLP', MLPClassifier()))
    basedModels.append(('SGD3000', SGDClassifier(max_iter=1000, tol=1e-4)))
    basedModels.append(('XGB_2000', xgb.XGBClassifier(n_estimators= 2000)))
    basedModels.append(('XGB_500', xgb.XGBClassifier(n_estimators= 500)))
    basedModels.append(('XGB_100', xgb.XGBClassifier(n_estimators= 100)))
    basedModels.append(('XGB_1000', xgb.XGBClassifier(n_estimators= 1000)))
    basedModels.append(('DTC' , DecisionTreeClassifier()))
    basedModels.append(('GBM'  , GradientBoostingClassifier(n_estimators=100,max_features='sqrt')))
    basedModels.append(('RF_Ent100'   , RandomForestClassifier(criterion='entropy',n_estimators=100)))
    basedModels.append(('RF_Gini100'   , RandomForestClassifier(criterion='gini',n_estimators=100)))
    basedModels.append(('ET100'   , ExtraTreesClassifier(n_estimators= 100)))
    basedModels.append(('ET500'   , ExtraTreesClassifier(n_estimators= 500)))
    basedModels.append(('ET1000'   , ExtraTreesClassifier(n_estimators= 1000)))

    basedModels.append(('LGBMC', LGBMClassifier()))


    return basedModels

# function for performing 10-fold cross validation of all the baseline models
def BasedLine2(X_train, y_train,models):
    # Test options and evaluation metric
    num_folds = 10
    scoring = 'accuracy'
    seed = 7
    results = []
```

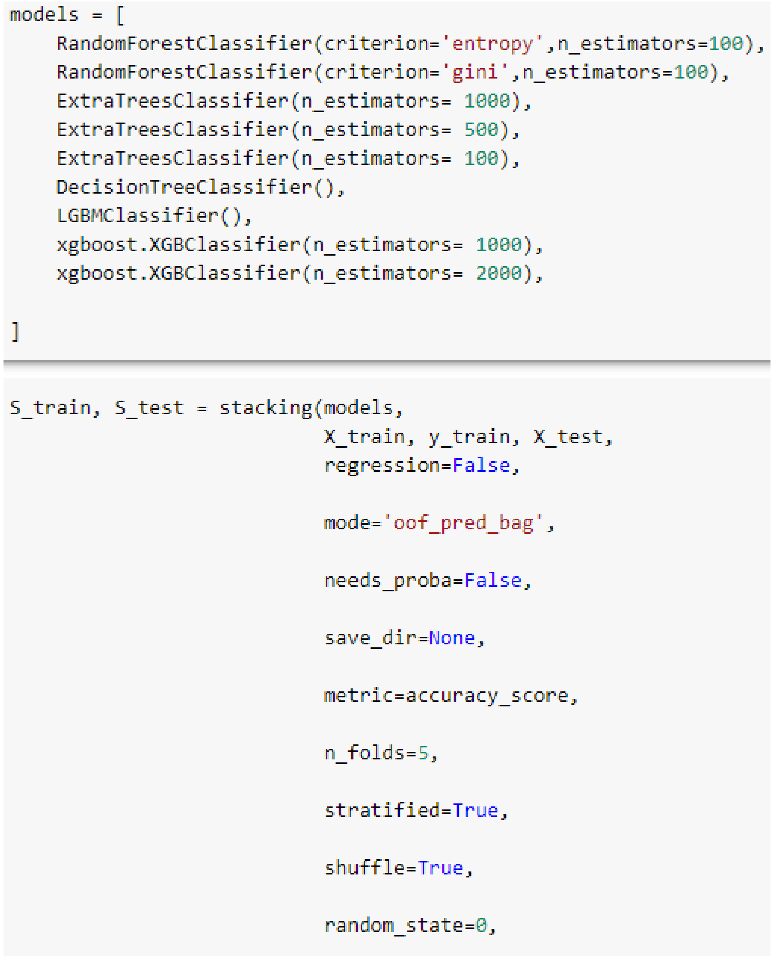
Code figure 9



```
models = GetBasedModel()
names,results = BasedLine2(X_train, y_train, models)
```

```
LR_L2: 0.899596 (0.030281)
LDA: 0.886368 (0.031440)
KNN7: 0.821684 (0.050636)
KNN5: 0.819088 (0.047286)
KNN9: 0.821702 (0.059193)
KNN11: 0.809789 (0.065414)
NB: 0.622211 (0.107128)
SVM Linear: 0.890368 (0.028855)
SVM RBF: 0.908860 (0.029878)
AB: 0.890386 (0.019525)
MLP: 0.911421 (0.027965)
SGD3000: 0.799281 (0.065580)
XGB_2000: 0.956351 (0.016948)
XGB_500: 0.948421 (0.023436)
XGB_100: 0.923333 (0.029639)
XGB_1000: 0.952386 (0.017121)
DTC: 0.955018 (0.019046)
GBM: 0.927298 (0.024003)
RF_Ent100: 0.959018 (0.020886)
RF_Gini100: 0.959000 (0.023319)
ET100: 0.960316 (0.020612)
ET500: 0.960316 (0.020612)
ET1000: 0.959000 (0.020131)
LGBMC: 0.955035 (0.021563)
```

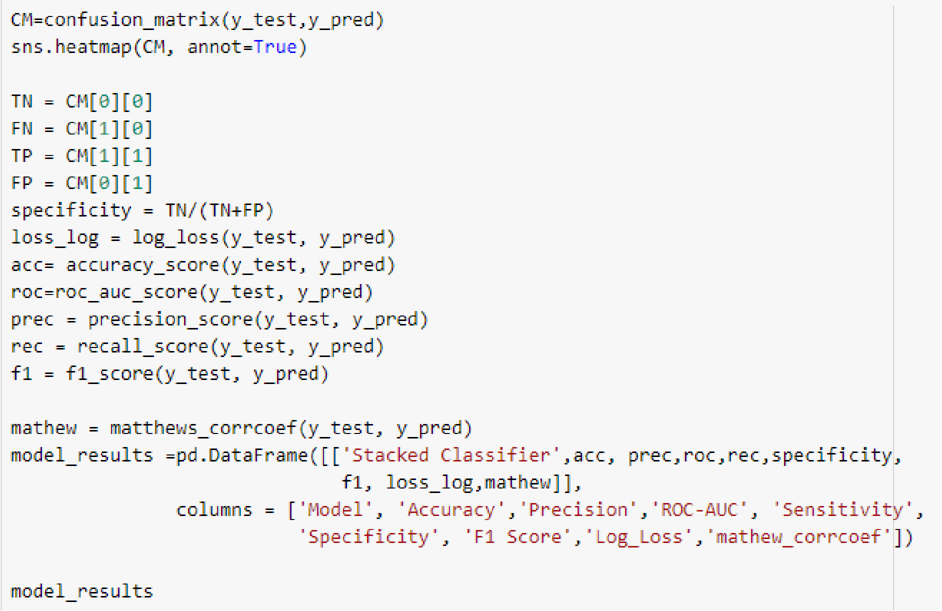
Code figure 10



```
models = [
    RandomForestClassifier(criterion='entropy',n_estimators=100),
    RandomForestClassifier(criterion='gini',n_estimators=100),
    ExtraTreesClassifier(n_estimators= 1000),
    ExtraTreesClassifier(n_estimators= 500),
    ExtraTreesClassifier(n_estimators= 100),
    DecisionTreeClassifier(),
    LGBMClassifier(),
    xgboost.XGBClassifier(n_estimators= 1000),
    xgboost.XGBClassifier(n_estimators= 2000),

]
```

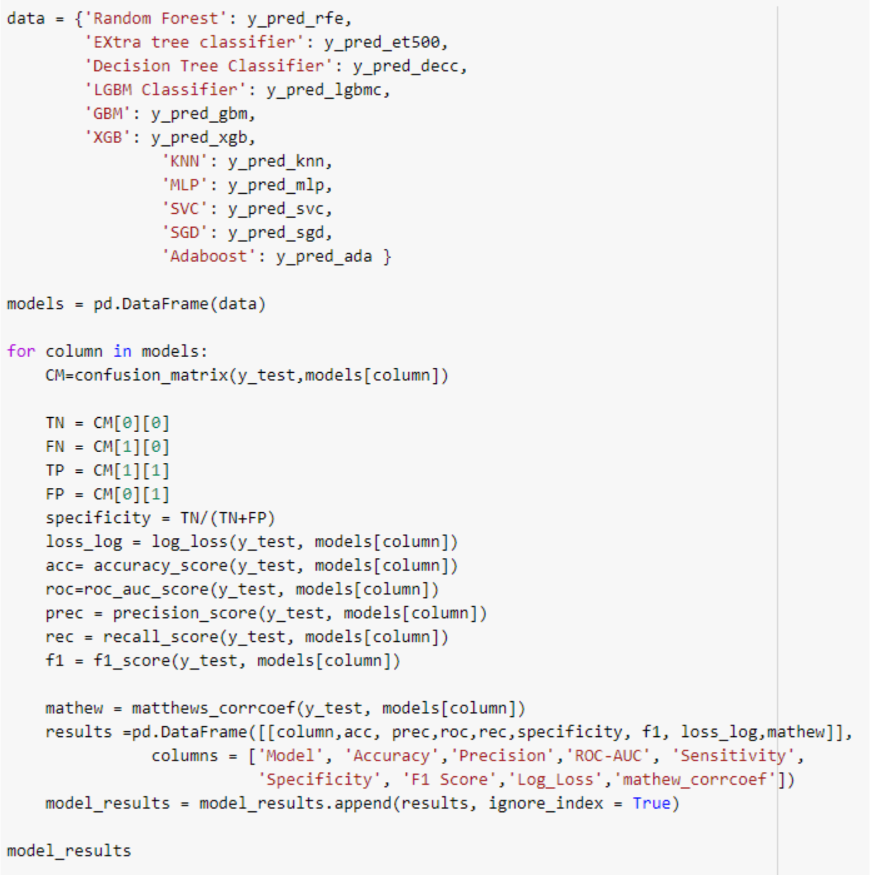
```
S_train, S_test = stacking(models,
                           X_train, y_train, X_test,
                           regression=False,

                           mode='oof_pred_bag',

                           needs_proba=False,

                           save_dir=None,

                           metric=accuracy_score,

                           n_folds=5,

                           stratified=True,

                           shuffle=True,

                           random_state=0,
```

Code figure 11



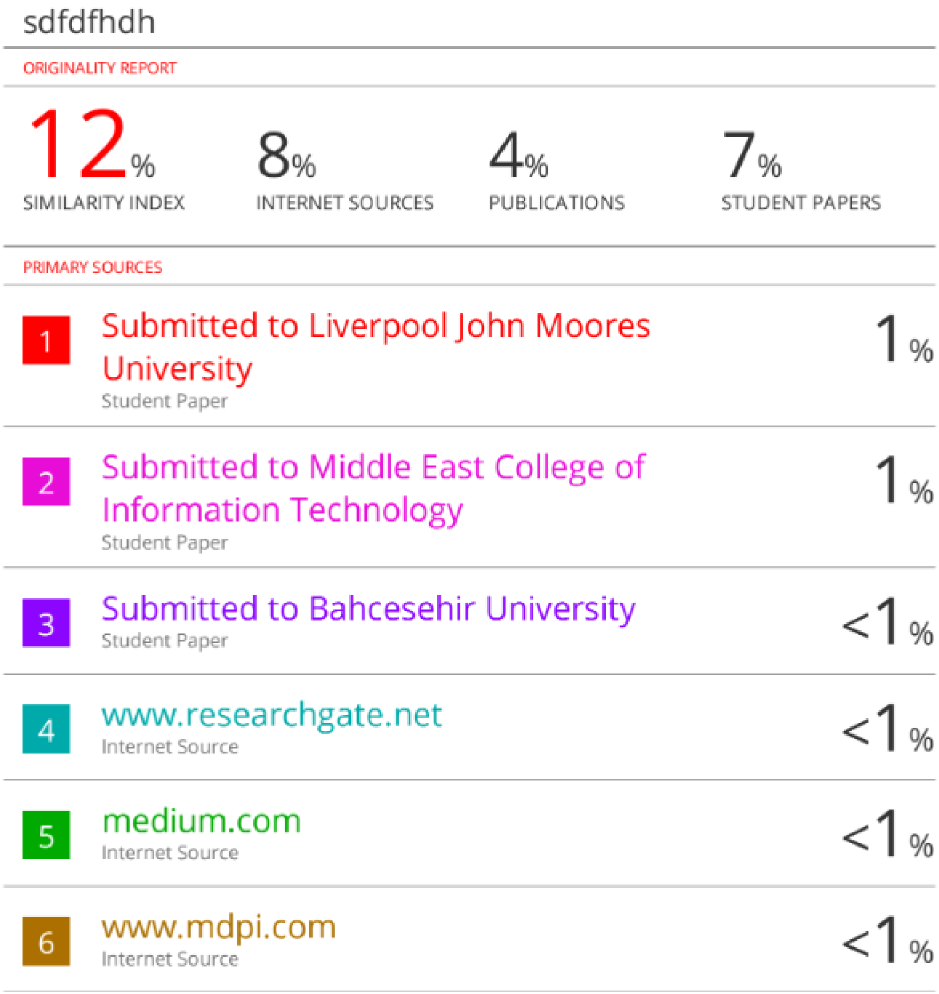
```
CM=confusion_matrix(y_test,y_pred)
sns.heatmap(CM, annot=True)

TN = CM[0][0]
FN = CM[1][0]
TP = CM[1][1]
FP = CM[0][1]
specificity = TN/(TN+FP)
loss_log = log_loss(y_test, y_pred)
acc= accuracy_score(y_test, y_pred)
roc=roc_auc_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

mathew = matthews_corrcoef(y_test, y_pred)
model_results =pd.DataFrame([['Stacked Classifier',acc, prec,roc,rec,specificity,
                              f1, loss_log,mathew]],
               columns = ['Model', 'Accuracy','Precision','ROC-AUC', 'Sensitivity',
                          'Specificity', 'F1 Score','Log_Loss','mathew_corrcoef'])

model_results
```

Code figure 12



```
data = {'Random Forest': y_pred_rfe,
        'EXtra tree classifier': y_pred_et500,
        'Decision Tree Classifier': y_pred_decc,
        'LGBM Classifier': y_pred_lgbmc,
        'GBM': y_pred_gbm,
        'XGB': y_pred_xgb,
                'KNN': y_pred_knn,
                'MLP': y_pred_mlp,
                'SVC': y_pred_svc,
                'SGD': y_pred_sgd,
                'Adaboost': y_pred_ada }

models = pd.DataFrame(data)

for column in models:
    CM=confusion_matrix(y_test,models[column])

    TN = CM[0][0]
    FN = CM[1][0]
    TP = CM[1][1]
    FP = CM[0][1]
    specificity = TN/(TN+FP)
    loss_log = log_loss(y_test, models[column])
    acc= accuracy_score(y_test, models[column])
    roc=roc_auc_score(y_test, models[column])
    prec = precision_score(y_test, models[column])
    rec = recall_score(y_test, models[column])
    f1 = f1_score(y_test, models[column])

    mathew = matthews_corrcoef(y_test, models[column])
    results =pd.DataFrame([[column,acc, prec,roc,rec,specificity, f1, loss_log,mathew]],
               columns = ['Model', 'Accuracy','Precision','ROC-AUC', 'Sensitivity',
                          'Specificity', 'F1 Score','Log_Loss','mathew_corrcoef'])
    model_results = model_results.append(results, ignore_index = True)

model_results
```

Code figure 13



sdfdfhdh

ORIGINALITY REPORT

12%	8%	4%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Liverpool John Moores University Student Paper	1%
2	Submitted to Middle East College of Information Technology Student Paper	1%
3	Submitted to Bahcesehir University Student Paper	<1%
4	www.researchgate.net Internet Source	<1%
5	medium.com Internet Source	<1%
6	www.mdpi.com Internet Source	<1%