PREDICTION OF DIABETES MELLITUS USING ENSEMBLE APPROACH

Sathyabama Institute of Science and Technology (Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of

Bachelor of Engineering Degree in

Computer Science and Engineering

By

CHEGU HEMA SAI SREE
(Reg. No.37110116)
CHITTALA SAHITHYA
(Reg.no.37110157)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

MARCH- 2021

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119
www.sathyabamauniversity.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **CHEGU HEMASAISREE (37110116) CHITTALA SAHITHYA (37110157)** who carried out the project entitled “**PREDICTION OF DIABETES MELLITUS USING ENSEMBLE APPROACH**” under my supervision from April 2020 to June 2020.

Internal Guide

Mrs. M.S. Roobini M.E., Ph.D

Head of the Department

Submitted for Viva voce Examination held on ____________________

Internal Examiner **External Examiner**

DECLARATION

I, **CHEGU HEMASAISREE (Reg.No:37110116) CHITTALA SAHITYA(37110157)**hereby declare that the Professional Training entitled **“PREDICTION OF DIABETES MELLITUS USING ENSEMBLE APPROACH”** done by me under The guidance of **Mrs. M.S. Roobini M.E., Ph.D**., **Dept of CSE** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D.**, **Dean**, School of Computing , **Dr.L.Lakshmanan M.E., Ph.D., and Dr. S. Vigneshwari M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mrs. M.S. Roobini M.E., Ph.D.;** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for completion of my project.

ABSTRACT

Diabetes is a kind of metabolic disease that forms by lack of insulin due to the malfunctioning of the pancreas. Diabetes can push a person into pathological destruction of pancreatic beta cells, coma, cardiovascular dysfunction, renal and retinal failure, joint failure, pathogenic effects on immunity, weight loss, and peripheral vascular diseases. So, for the early detection of diabetes, a robust framework was proposed, where outlier rejection, filling the missing values, data standardization, K-fold validation, and different Machine Learning (ML) classifiers (k-NN, decision trees (DT), random forest (RF), naive Bayes (NB), XGboost and Adaboost were used. To improve the result, the weighted ensembling of different ML models also proposed here. The corresponding Area Under ROC Curve (AUC) of the ML model as the performance metric estimated these weights. Using the grid search technique, the AUC is then maximized during hyperparameter tuning. All experiments were conducted under the same experimental conditions on publicly available Diabetes dataset population near Phoenix, Arizona of 768 female diabetic patients, where there are 268 diabetic patients (positive) and 500 non-diabetic patients (negative) with eight different attributes

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Diabetes is a very dangerous disease and does not able to cure. If this disease affects once, it will maintain in your life time. At the same time, your blood having too much of glucose can cause health issues. Like kidney disease, heart disease, stroke, eye problems, dental disease, foot problems, nerve damage.so you can take step to oversee your diabetes and avert these complications. With the development of living standards, diabetes is increasingly common in people’s daily life. Therefore, how to quickly and accurately diagnose and analyze diabetes is a topic worthy studying. In medicine, the diagnosis of diabetes is according to fasting blood glucose, glucose tolerance, and random blood glucose levels. The earlier diagnosis is obtained, the much easier we can control it. Machine learning can help people make a preliminary judgment about diabetes mellitus according to their daily physical examination data, and it can serve as a reference for doctors. For machine learning method, how to select the valid features and the correct classifier are the most important problems. The familiar types of diabetes.

1.1 Type 1 diabetes

1.2 Type 2 diabetes

1.3 Gestational diabetes

1.1 TYPE 1 DIABETES:

Prediabetes is a condition of sufficiently elevated blood glucose levels, although not as severe as normal, to be listed as diabetes. The cells that contain insulin of the pancreas of the body are type-1 diabetes and kill more than 90% of them. Just about 5 to 10 percent of the individuals with diabetes have type1. Diabetes may contribute to lifelong damage and dysfunction of organs, including the skin, liver, ears, blood vessel.

or Latin American ethnicity residing in the United States.

1.2 TYPE 2 DIABETES:

Type 2 diabetes may be graded (T2D). Patients of type 1 diabetes are typically younger, sometimes fewer than 30 years of age. High appetite, frequent urination and elevated blood pressure are common symptoms of health. A type of diabetes cannot be effectively controlled only with oral medicines, such that insulin therapy is required for patients. Type 2 diabetes is common among the middle-aged and elderly, frequently combined with obesity, elevated blood pressure, dyslipidaemia, atherosclerosis and other diseases. Type 2 diabetes is largely preventable by staying a normal weight, exercising regularly, and eating properly. Treatment involves exercise and dietary changes. If blood sugar levels are not adequately lowered, the medication metformin is typically recommended. Many people may eventually also require insulin injections. In those on insulin, routinely checking blood sugar levels is advised; however, this may not be needed in those taking pills. Bariatric surgery often improves diabetes in those who are obese.

1.3 GESTATIONAL DIABETES:

Gestational diabetes is diabetes diagnosed for the first time during pregnancy (gestation). Like other types of diabetes, gestational diabetes affects how your cells use sugar (glucose). Gestational diabetes causes high blood sugar that can affect your pregnancy and your baby's health.

While any pregnancy complication is concerning, there's good news. Expectant mothers can help control gestational diabetes by eating healthy foods, exercising and, if necessary, taking medication. Controlling blood sugar can keep you and your baby healthy and prevent a difficult delivery.

1.4 SYSTEM CONFIGURATION:

- **H/W SYSTEM CONFIGURATION**

System	:	Intel core I3.
Hard Disk	:	120 GB
Monitor	:	15"LED
Input Devices	:	Keyboard,
Mouse Ram	:	4GB

- **S/W SYSTEM CONFIGURATION**

Operating system	:	Windows 10
Coding Language	:	Python
Toolkit	:	Jupyter notebook
Database	:	Diabetes data set

1.5 PROBLEM DEFINITION:

Many algorithms, such as the supporting vector machine (SVM), decision tree (DT), technical regression, have been used lately to predict diabetes, including traditional machine learning. Weighted the least squares provided by the vector machine(WLS-SVM) in order to forecast type 2 diabetes, with the algorithm used for quantum particle swarm Optimization (QPSO). Linear Discriminant Analysis (LDA) has been used by writers to minimise estimation and separate the functions. Built Prediction Models for multiple Type 2 diabetes prediction events to tackle high-dimensional data sets on the basis of logistic regression. Concentrated on glucose and used support vector regression (SVR) to model diabetes as a multivariate regression issue. Set methods were used to improve accuracy, however, by more and more experiments. Suggested new ensemble technique, which involves 30 methods of computer education, named rotational wood.

CHAPTER 2

LITREATURE SURVEY

K. Vijay Kumar, and others [11] proposed random Forest algorithm for the prediction of diabetes to develop a system that can predict early diabetes for a more accurately-focused patient by using the machine-learning Random Forest algorithm. The model suggested provides the best results in diabetic prediction and the results demonstrated that the prediction system can efficiently, efficiently and most importantly, immediately predict diabetes disease.

The predictive onset of diabetes was presented to Nanos Nnamoko et al [13]. A supervised learning approach is used to combine their results with five widely used classifications. The results will be presented and compared to similar studies using the same dataset in the book. Diabetes beginning can be predicted with greater accuracy by using the proposed method. Roof N. The goal is to avoid diabetes by three numerous regulated methods of learning, namely SVM, logistic regression, and ANN, proposed by Joshi et al [11]. Predicting Diabetes Prediction Using Machine Learning Techniques. The project advocates an effective method for early diabetes detection.

Deeraj Shetty et al [15] suggested prediction of diabetic disease using data extraction, assembling a prediction method of intelligent diabetes that offers analysis to patients with diabetes via a database of diabetes. In this system, algorithms such as Bayesian and KNN are proposed to be applied to the database of patients with diabets and analysed by taking various diabetes attributes for diabetes prediction.

In order to boost diabetes diagnostics precision, T.Santhanametal[1] suggests a system that uses K-means, Genetic algorithms and SVM. Data cleaning was accomplished by replacementing sing-values for mean. K-means used to remove data from the helpful vector machine, and genetic algorithms for the optimum identifying eatures (SVM). In each run, GA selected different attributes and the rating accuracy was recorded from the original set of attributes. The experiment was replicated 50 times to obtain a stable outcome and the findings were seen in the table. The experimental findings indicate that the suggested model has an average accuracy of 98,79 per cent for Pima Indian diabetes in the UCI archive. In contrast to the SVM clustered K-means Data Preparation Scheme, stronger findings have also been obtained in the proposed approach (96.71 percent).

We are currently evaluating the same data packet split into the same training and prediction sets, both using a logistic regression and a linear perceptron model[5]. We expect to use all the threematics - ADAP, logistic regression and a linear perceptron to compare the ROC curves for this forecast. This paper is used by the amalgamated approach and obtained with greater accurate results in the compact formation of uncertainty matrix as a linear regression Data mining is a method that gathers valuable knowledge from a vast quantity of information that helps you to obtain more information. The use of data mining tools and methods thus leads to the diabetes prediction and thus decreases costs for care. In order to diagnose diabetes and discover effective approaches to also manage them, data mining methods are used in the field of medicinal data.

In this article[6], we used the neighbouring algorithm K-nearest to diagnose diabetes mellitus. The findings revealed that, as k value rises, accuracy and error rates have improve. We measured the correctness and error rate of K = 3,5. KNN is one of the most powerful and commonly diagnosed artificial intelligence algorithms. many classification methods have been used in the data mining process. More precise and effective results are obtainable via KNN. These methods were used to split the data into separate sets such that a relationship could be effectively defined with different attributes. Other data mining techniques include kernel density.

we use three different classificational algorithms that are used in this experiment: ZeroR, OneR and Naiv. Different data mining techniques are used to help health professionals diagnose diabetes. This experiment reveals the quickest Naïve Bayes and the slowest ZeroR. A faulty data mining technique is used to identify efficiency comparisons. Each model shall be transformed into rules and these rules shall be incorporated. This study article deals with machine learning, which has a sophisticated data structure and an extensive epidemiological and genetic diabetes risk base to revolutionise the estimation of diabetes risk. This paper addresses Machine Learning. Diabetes diagnosis is the answer to recovery in the early stages. This dissertation outlines a machine-learning solution to diabetes prediction. The methodology can also help researchers build a precise and efficient method to help them make better decisions on the disease status of clinicians.

Diabetes in young people and ancient cultures was influenced by Amina Azar et al. in [7]. These are raised day by day and it is not curable. For early prediction, data mining is used. This paper in key purpose is offers the distinction and recommend best algorithm. You will use the PID datasets. In order to predict early diabetes diagnosis in highest precision and performance, decisions tree, Naive Bayes and K-Nearest neighbouring algorithms are contrasted and used. The WEKA is used for testing and validation using rapid miner. The decision tree is the perfect algorithm for predictions. It offers 75.65 percent of the precision.

CHAPTER 3

AIM AND SCOPE

3.1 AIM:

Diabetes is a group of metabolic disorders charaterzied by abnormal metabolism which results most notably in hyperglycemia due to defects in insulin secreation, insulin action or both. Diabets is a serious chronic disease without a cure and it is associated with significant morbidity and mortality. Diabetes is a serious disesae associated with acute(due to hyperglycemia) and chronic(due to vascular damages)complications.

3.2 OBJECTIVE:

Comparing to the existing models our proposed work gives us a better effiency. So, in this analysis, Machine Learning algorithms have been used to classify diabetes.The extensive experiments for the selection of the best performing feature selection methods with selected attribute numbers.

3.3 SCOPE:

So, for the early detection of diabetes, a robust framework was proposed, where outlier rejection, filling the missing values, data standardization, K-fold validation, and different Machine Learning (ML) classifiers (k-NN, random forest (RF),and Xgboost (XB).Adaboost were used.To improve the result, the weighted ensembling of different ML models also proposed here.The corresponding Area Under ROC Curve (AUC) of the ML model as the performance metric estimated these weights. Algorithms used in this are.

3.3.1 Knn

3.3.2 Random Forest

3.3.3 Adaboost

3.3.4 Xgboost

3.3.1 knn:

Neighbor is often considered a lazy learning algorithm classifying datasets dependent on their similarities. "K" is the number of dataset objects for classification considered. Both regression and classification issues are caused by KNN. This approach categorises new objects on the basis of tests for similarity. It's very quick and intuitive, certain benefits of KNN. Good classification if there is an amount of samples. KNN has even some inconveniences, such as the importance of K. The test step of K- Nearest neighbour (KNN) is computer-compliant. It needs several samples for accuracy.

3.3.2 Random Forest :

This classification algorithm is equivalent to the classification system for ensemble learning. The recovery and other activities are achieved by constructing a community of decision trees at the training level and during class performance, which may be the way individual trees are categorised or predict regression. This consistency in the classification of decision-making treasures requires overfitting data.

3.3.3 Adaboost:

Basically, weak models are added sequentially, trained using the weighted training data. Generally, the process continues until a pre-set number of weak learners have been created. Once completed, you are left with a pool of weak learners each with a stage value. Predictions are made by calculating the weighted average of the weak classifiers.

or -1.0. The predicted values are weighted by each weak learners stage value. The prediction for the ensemble model is taken as a sum of the weighted predictions. If the sum is positive, then the first class is predicted, if negative the second class is predicted.

3.3.4 Xgboost:

Study recently discovered a 'XGBoost' algorithm and its usage is very useful for classification of machine learning. The performance is even faster and easier, because it is the implementation of a strengthened decision tree. This classification model is used for optimising the model's efficiency and tempo.

XGBoost initially started as a research project by Tianqi Chen as part of the Distributed (Deep) Machine Learning Community (DMLC) group. Initially, it began as a terminal application which could be configured using a libsvm configuration file. It became well known in the ML competition circles after its use in the winning solution of the Higgs Machine Learning Challenge. Soon after, the Python and R packages were built, and XGBoost now has package implementations for Java, Scala, Julia, Perl, and other languages. This brought the library to more developers and contributed to its popularity among the Kaggle community, where it has been used for a large number of competitions.

CHAPTER 4
SYSTEM ANALYSIS

4.1 EXISTING SYSTEM:

Dataset is tremendously in volume, colossal in the assortment, to remove helpful data to settle on business choice or finding the comparative examples to settle on a better choice. It is utilized to find new examples, find comparable connections among information, co-relations between information, this can find answers for the issues, creating rules from old information, settling on best choices of ad lib the business arrangements, finding concealed information design from leaving datasets, expectation of future yield, i.e. practices and patterns. which is utilized to fabricate another model from the input dataset. It tackles the issue of ordering the dataset and doling out the class marks for the informational index. An arrangement system examining the informational index and predicts the class names or allocates the gathering mark. The key goal of characterization is to produce the new models with great speculation anticipating capacity. The new model ought to be well form model to precisely characterize the dataset on their qualities to anticipate class names. Order demonstrates which takes the occasion of the dataset and doles out to the specific class name.

The development of long-term complications is influenced by hyperglycaernia. Poor control of diabetes accelerates their progression. Thus, to prevent complications, good control of diabetes is essential and the management of diabetes should therefore aim to improve glycaemic control beyond that required to control its symptoms. Intensified therapy and maintaining near-normal blood glucose levels can result in considerable reduction in the risk of development of retinopathy.

availability of successful prevention strategies, essential health care requirements and facilities for self-care are often inadequate in this Region. Action is needed at all levels of health care and in the various aspects of diabetes care to bridge this gap and to improve health care delivery to people with diabetes. Education of the health care team on the management of diabetes and on how to educate people with diabetes is one major aspect that requires strengthening. Even though resources vary widely within the Region, the primary resource in diabetes care is now recognized to be the people with diabetes themselves, supported by well trained and enthusiastic health care professionals. This resource can be strengthened nearly everywhere by education.

4.2 PROPOSED SYSTEM:

4.2.1 ***Diabetes dataset:*** Reading the data from diabetes dataset(here the dataset is present in csv files)

4.2.2 ***Prepocessing****:* preprocessing is used to process the data weather there is a null value in the data set before implementing the algorithms.

4.2.3 ***Train data***: The sample of data used to fit the model.

The actual dataset that we use to train the model (weights and biases of neural network) the model *sees* and *learns* from this data.

In train data we use two types:

4.2.3.1 Grid search: is the process of scanning the data for a given model. Depending on the type of model utilized, certain parameters necessary

4.2.3.2 Hyper Parameters: hyperparameter is a parameter whose value is used to control the learning process.

4.2.4 ***Test data:*** The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

Step1: Input the diabetic dataset.

Step2: The dataset has been randomly split into training and testing data.

Step3: Applying the machine learning algorithms to predicting the diabetic disease.

Step4: Finally the performances of classifiers are evaulated.

CHAPTER 5

DESIGN AND TESTING

5.1 SYSTEM ARCHITECTURE:

In the PIMA Indians Diabetes Data Collection, we suggest a new diabetes prediction pipeline. Preprocessing is the heart of obtaining the most advanced outcome in the proposed

pipeline, which is the outer refuse, Filling missed values, standard info, collection of features and cross-validation of K-fold. We see the average value in the missing attribute place rather than median value, since it is more central to the average of the distribution of the attribute. Cross- validation of the dataset is carried out with caution to retain the percentage of the class proportion in the same manner as in the initial dataset. Various ML (k-nearest Neighbor (kNN) grouping modules is RF, DT, NB, AB

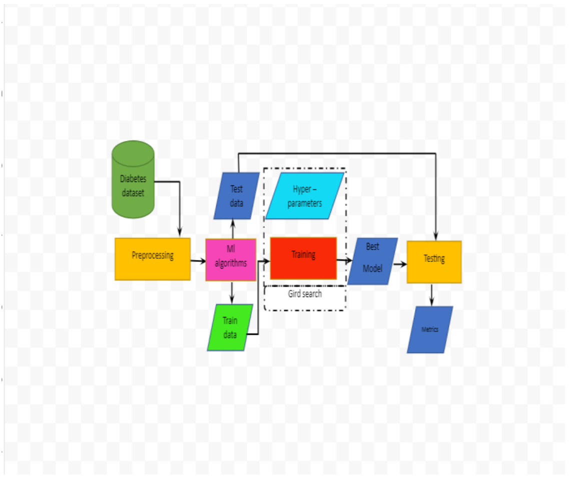


Fig.no.5.1 Architecture

and XGBoast (XB)). Built in our pipeline suggested.

5.2 DATA FLOW DIAGRAM:

- The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
- The data flow diagram (DFD) is one of the most important modelling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
- DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
- DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

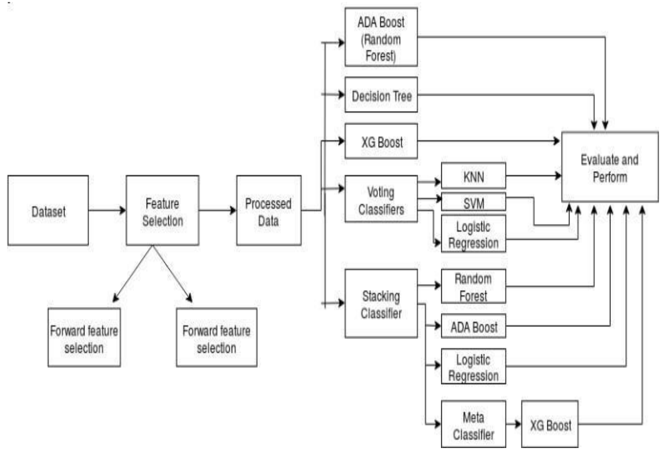


Fig.no.5.2 Data Flow Diagram

5.3 UML DIAGRAMS:

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

5.3.1 goals:

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.

- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

5.3.2 use case diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

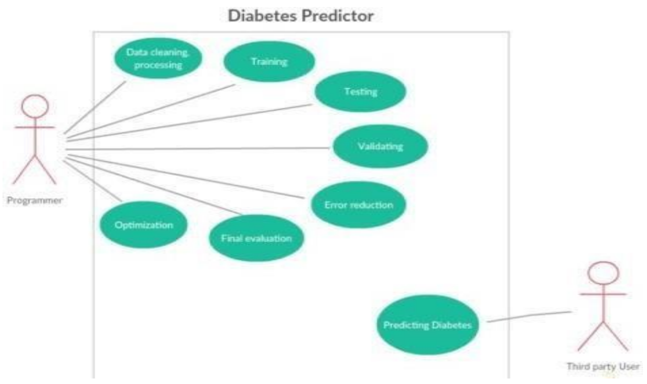


Fig.no.5.3.2 use case diagram

5.3.3 class diagram:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information. In the design of a system, a number of classes are identified and grouped together in a class diagram that helps to determine the static relations between them. With detailed modeling, the classes of the conceptual design are often split into a number of subclasses.

In order to further describe the behavior of systems, these class diagrams can be complemented by a state diagram or UML state machine

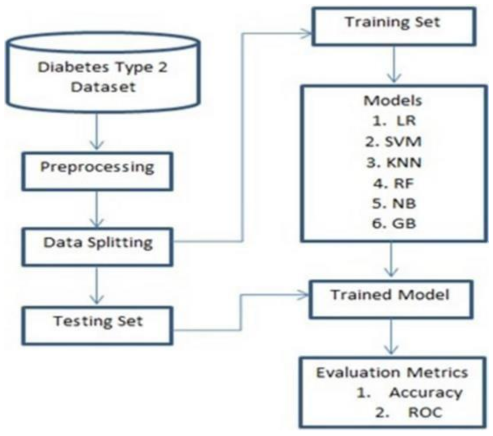


Fig.no.5.3.3 class diagram

5.4 SYSTEM TESTING:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

There are 3 different types of testing

5.4.1 Unit testing

5.4.2 Integration testing

5.4.3 Acceptance testing

5.4.4 White box testing

5.4.5 Black box testing

5.4.1 Unit testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results. Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

5.4.2 Integration testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields.

combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components. Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error. Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.
The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.
Test Results All the test cases mentioned above passed successfully. No defects encountered.
Integration tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.
Integration testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
InvalidInput	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.

5.4.3 Acceptance testing:
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.
Test Results All the test cases mentioned above passed successfully. No defects encountered.

In engineering and its various subdisciplines, acceptance testing is a test conducted to determine if the requirements of a specification or contract are met. It may involve chemical tests, physical tests, or performance tests.

In systems engineering, it may involve black-box testing performed on a system (for example: a piece of software, lots of manufactured mechanical parts, or batches of chemical products) prior to its delivery.

In software testing, the ISTQB defines *acceptance testing* as:

Formal testing with respect to user needs, requirements, and business processes conducted to determine whether a system satisfies the acceptance criteria and the enable the user, customers or other authorized entity to determine whether to accept the system.

—Standard Glossary of Terms used in Software Testing

Acceptance testing is also known as user acceptance testing (UAT), end-user testing, operational acceptance testing (OAT), acceptance test-driven development (ATDD) or field (acceptance) testing. Acceptance criteria are the criteria that a system or component must satisfy in order to be accepted by a user, customer, or other authorized entity.

5.4.4 White box testing:

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or atleast its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level. White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of software testing that tests internal structures or workings of an application, as opposed to its functionality.

to design test cases. The tester chooses inputs to exercise paths through the code and determine the expected outputs. This is analogous to testing nodes in. Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements. Where white-box testing is design-driven, that is, driven *exclusively* by agreed specifications of how each component of software is required to behave (as in DO-178C and ISO 26262 processes) then white-box test techniques can accomplish assessment for unimplemented or missing requirements.

5.4.5 Black box testing:

Black Box Testing is testing the software without any knowledge of theinner workings, structure or language of the module being tested. Black boxtests, as most other kinds of tests, must be written from a definitive sourcedocument, such as specification or requirements document, such asspecification or requirements document. It is a testing in which the softwareunder test is treated, as a black box . You cannot “see” into it. The test providesinputs and responds to outputs without considering how the software works. Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. It is sometimes referred to as specification-based testing.

CHAPTER 6

SUMMARY AND CONCLUSION

Diabetes mellitus (DM), is a series of metabolic disorders in human body due to high level of blood sugar in body. If left untreated, it can cause many complications in the long run. Diabetes is majorly caused due to dis-functioning of pancreas leading to failure in production of required insulin. From the above survey its understood thatitis always better to use more than one classifier for predicting the output as its accuracy is higher than accuracy of single classifier. Bagging ensemble classifier should be used in the predicting System so that can obtain better and efficient results. Diabetes mellitus (DM), is a series of metabolic disorders in human body due to high level of blood sugar in body. If left untreated, it can cause many complications in the long run. Diabetes is majorly caused due to dis-functioning of pancreas leading to failure in production of required insulin. From the above survey its understood thatitis always better to use more than one classifier for predicting the output as its accuracy is higher than accuracy of single classifier. Bagging ensemble classifier should be used in the predicting System so that can obtain better and efficient results.

REFERENCES

1. K. Vijiya Kumar, B. Lavanya, I. Nirmala, S. Sofia Caroline, &quot;Random Forest Algorithm for the Prediction of Diabetes &quot;. Proceeding of International Conference on Systems Computation Automation and Networking, 2019.

2. Nonso Nnamoko, Abir Hussain, David England, &quot;Predicting Diabetes Onset: An Ensemble Supervised Learning Approach &quot;. IEEE Congress on Evolutionary Computation (CEC), 2018.

3. Deeraj Shetty, Kishor Rit, Sohail Shaikh, Nikita Patil, &quot;Diabetes Disease Prediction Using Data Mining “. International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), 2017.

4. T. Santhanama, M.S Padmavathi b. “Application of K-Means and Genetic Algorithms for Dimension Reduction by Integrating SVM for Diabetes Diagnos is”. Procedia Computer Science, vol.47, pp.76-83,2015.

5. Karnika Dwivedi, Dr. Hari Om Sharan.2018.Review on Prediction of Diabetes Mellitus using Data Mining Technique fromInternational Journal of Engineering and Technical Research (IJETR).

6. Amina Azar,Yasir Ali, Muhammad Awais, KhurramZaheer,“Data Mining Models Comparison for Diabetes Prediction”, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 9, No. 8, 2018.

7. Amina Azar,Yasir Ali, Muhammad Awais, KhurramZaheer,“Data Mining Models Comparison for Diabetes Prediction”, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 9, No. 8, 2018

8. V. Vijayan, A. Ravikumar

International Journal of Computer Applications (17) (2014), p. 95
9V.V. Vijayan, C. Anjali
Prediction and diagnosis of diabetes mellitus—A machine learning approach
IEEE Recent Advances in Intelligent Computational Systems (RAICS) (2015), pp. 122-127

10.D. Verma, N. MishraAnalysis and prediction of breast cancer and diabetes disease datasets using data mining classification techniquesProceedings of the international conferenceonintelligentsustainablesystems (ICISS) (2017), pp. 533-538

11. B. Sudharsan, M. Peeples, M. Shomali Hypoglycemia prediction using machine learning models for patients with type 2 diabetes.

12 E. Standl, K. Khunti, T.B. Hansen, O. SchnellTheglobalepidemicsofdiabetesinthe 21stcentury:Current situationandperspectivesEuropeanJournalof Preventive Cardiology, 26 (2_suppl) (2019), pp.

13.A new artificial neural networks approach for diagnosing diabetes disease type IInternational Journal of Advanced Computer Science and Applications, 7 (2016), pp. 89-94

14A.S. Rani, S. JyothiPerformance analysis of classification algorithms under different datasetsProceedings of the 3rd international conference on computing for sustainable global development (INDIACom) (2016), pp. 1584-1589

15.R. QuinlanC4.5: Programs for machine learningMorgan Kaufmann Publishers, San Mateo, Calif (1993)

APPENDIX

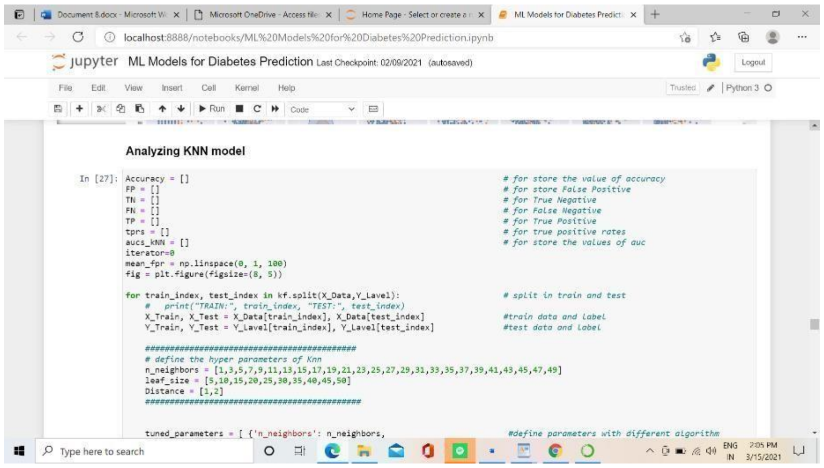
A.SOURCE CODE:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib     inline
import seaborn as sns
# sns.set(style="whitegrid")
import warnings
from sklearn.svm import SVC, NuSVC
warnings.filterwarnings('ignore')
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.neural_network import MLPClassifier
import xgboost as xgb
from scipy import stats
from scipy.stats import uniform, randint
from sklearn.metrics import f1_score
from sklearn.model_selection import KFold, StratifiedKFold, RepeatedStratifiedKFold
from sklearn.metrics import roc_curve, auc, accuracy_score
```

```
# from tflearn.data_utils import to_categorical
from sklearn import preprocessing
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import classification_report
from scipy import interp
from sklearn.metrics import confusion_matrix
from sklearn.decomposition import PCA from
sklearn.decomposition  import  FastICA  from
keras.utils import to_categorical
Renamed_feature= []               #list of names that will rename to feature column
all_clf_res=[]                  #every   classifier   auc   values   are   stored   in   it
random_initializer=100            #random initializer
n_dots=50
for i in range(8):
  #for renaming dataset of columns features F1 -- F8
  Renamed_feature.append('F'+str(i+1))
Parameters :
  Input -
  data is the pandas type variable iqr_Mean
  - for outleir rejection with Mean
  iqr_Medain- for outleir rejection with Medain
  iqr- for drop the outleir
  manual -for manual rejection
  Return - dataframe with outleir rejection
  filled with Input parameter
```

B.SCREEN SHOTS:

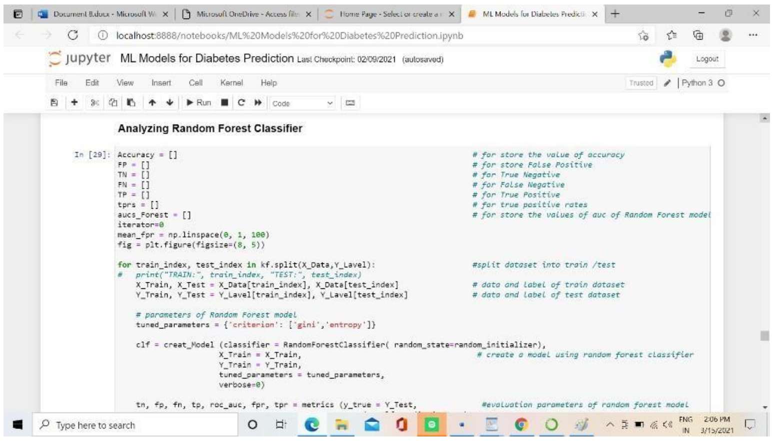
SOURCE CODE:


localhost:8888/notebooks/ML%20Models%20for%20Diabetes%20Prediction.ipynb
jupyter ML Models for Diabetes Prediction Last Checkpoint: 02/09/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
Type here to search

Analyzing KNN model

```
In [27]: Accuracy = []                                                        # for store the value of accuracy
         FP = []                                                              # for store False Positive
         TN = []                                                              # for True Negative
         FN = []                                                              # for False Negative
         TP = []                                                              # for True Positive
         tprs = []                                                            # for true positive rates
         aucs_kNN = []                                                        # for store the values of auc
         iterator=0
         mean_fpr = np.linspace(0, 1, 100)
         fig = plt.figure(figsize=(8, 5))

         for train_index, test_index in kf.split(X_Data,Y_Lavel):             # split in train and test
             #   print("TRAIN:", train_index, "TEST:", test_index)
             X_Train, X_Test = X_Data[train_index], X_Data[test_index]        #train data and label
             Y_Train, Y_Test = Y_Lavel[train_index], Y_Lavel[test_index]      #test data and label

             ###########################################
             # define the hyper parameters of Knn
             n_neighbors = [1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,35,37,39,41,43,45,47,49]
             leaf_size = [5,10,15,20,25,30,35,40,45,50]
             Distance = [1,2]
             ############################################


             tuned_parameters = [ {'n_neighbors': n_neighbors,                 #define parameters with different algorithm
```

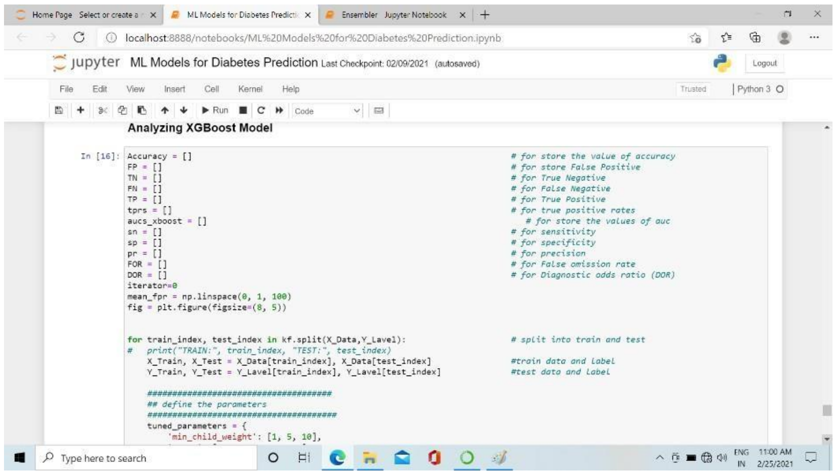
Fig.no.B.1.1:PYTHON CODE


localhost:8888/notebooks/ML%20Models%20for%20Diabetes%20Prediction.ipynb
jupyter ML Models for Diabetes Prediction Last Checkpoint: 02/09/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
Type here to search

Analyzing Random Forest Classifier

```
In [29]: Accuracy = []                                                        # for store the value of accuracy
         FP = []                                                              # for store False Positive
         TN = []                                                              # for True Negative
         FN = []                                                              # for False Negative
         TP = []                                                              # for True Positive
         tprs = []                                                            # for true positive rates
         aucs_Forest = []                                                     # for store the values of auc of Random Forest model
         iterator=0
         mean_fpr = np.linspace(0, 1, 100)
         fig = plt.figure(figsize=(8, 5))

         for train_index, test_index in kf.split(X_Data,Y_Lavel):             #split dataset into train /test
         #   print("TRAIN:", train_index, "TEST:", test_index)
             X_Train, X_Test = X_Data[train_index], X_Data[test_index]        # data and label of train dataset
             Y_Train, Y_Test = Y_Lavel[train_index], Y_Lavel[test_index]      # data and label of test dataset

             # parameters of Random Forest model
             tuned_parameters = {'criterion': ['gini','entropy']}

             clf = creat_Model (classifier = RandomForestClassifier( random_state=random_initializer),
                                X_Train = X_Train,                             # create a model using random forest classifier
                                Y_Train = Y_Train,
                                tuned_parameters = tuned_parameters,
                                verbose=0)

             tn, fp, fn, tp, roc_auc, fpr, tpr = metrics (y_true = Y_Test,      #evaluation parameters of random forest model
```

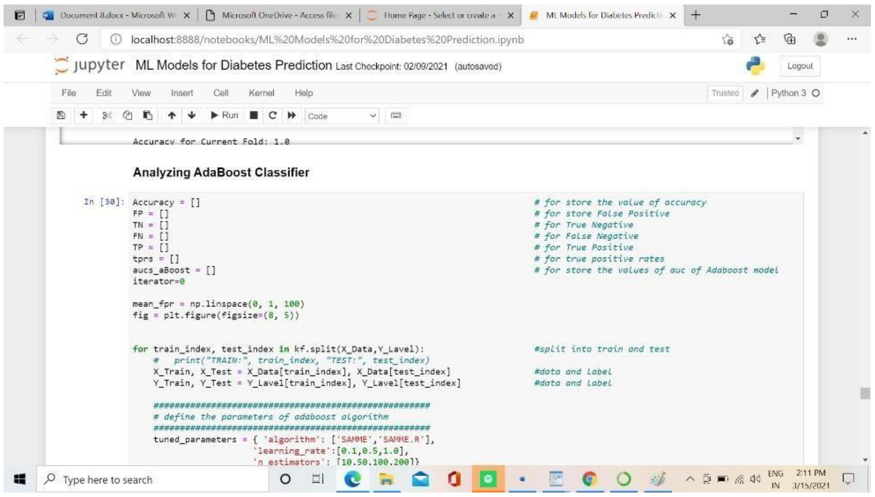
Fig.no.B.1.2:PYTHON CODE


Home Page Select or create a
ML Models for Diabetes Predicti
Ensembler Jupyter Notebook
localhost:8888/notebooks/ML%20Models%20for%20Diabetes%20Prediction.ipynb
jupyter ML Models for Diabetes Prediction Last Checkpoint: 02/09/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
Type here to search
ENG IN 11:00 AM 2/25/2021

Analyzing XGBoost Model

```
In [16]: Accuracy = []                                                          # for store the value of accuracy
         FP = []                                                                # for store False Positive
         TN = []                                                                # for True Negative
         FN = []                                                                # for False Negative
         TP = []                                                                # for True Positive
         tprs = []                                                              # for true positive rates
         aucs_xboost = []                                                          # for store the values of auc
         sn = []                                                                # for sensitivity
         sp = []                                                                # for specificity
         pr = []                                                                # for precision
         FOR = []                                                               # for False omission rate
         DOR = []                                                               # for Diagnostic odds ratio (DOR)
         iterator=0
         mean_fpr = np.linspace(0, 1, 100)
         fig = plt.figure(figsize=(8, 5))


         for train_index, test_index in kf.split(X_Data,Y_Lavel):               # split into train and test
         #   print("TRAIN:", train_index, "TEST:", test_index)
             X_Train, X_Test = X_Data[train_index], X_Data[test_index]          #train data and Label
             Y_Train, Y_Test = Y_Lavel[train_index], Y_Lavel[test_index]        #test data and Label

             ######################################
             ## define the parameters
             #######################################
             tuned_parameters = {
                 'min_child_weight': [1, 5, 10],
```

Fig.no.B.1.3 PYTHON CODE


Document 8.docx - Microsoft W
Microsoft OneDrive - Access file
Home Page - Select or create a
ML Models for Diabetes Predicti
localhost:8888/notebooks/ML%20Models%20for%20Diabetes%20Prediction.ipynb
jupyter ML Models for Diabetes Prediction Last Checkpoint: 02/09/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
Type here to search
ENG IN 2:11 PM 3/15/2021

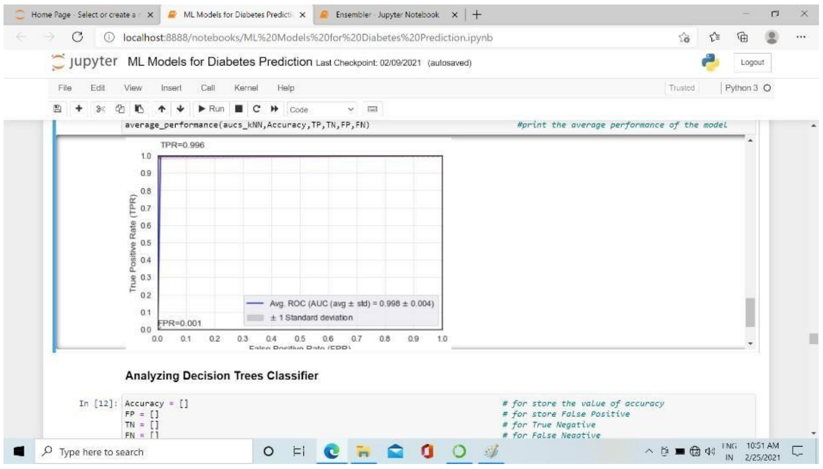
```
Accuracy for Current Fold: 1.0
```

Analyzing AdaBoost Classifier

```
In [30]: Accuracy = []                                                          # for store the value of accuracy
         FP = []                                                                # for store False Positive
         TN = []                                                                # for True Negative
         FN = []                                                                # for False Negative
         TP = []                                                                # for True Positive
         tprs = []                                                              # for true positive rates
         aucs_aBoost = []                                                       # for store the values of auc of Adaboost model
         iterator=0

         mean_fpr = np.linspace(0, 1, 100)
         fig = plt.figure(figsize=(8, 5))


         for train_index, test_index in kf.split(X_Data,Y_Lavel):               #split into train and test
             #   print("TRAIN:", train_index, "TEST:", test_index)
             X_Train, X_Test = X_Data[train_index], X_Data[test_index]          #data and Label
             Y_Train, Y_Test = Y_Lavel[train_index], Y_Lavel[test_index]        #data and Label

             #####################################################
             # define the parameters of adaboost algorithm
             #####################################################
             tuned_parameters = { 'algorithm': ['SAMME','SAMME.R'],
                                  'learning_rate':[0.1,0.5,1.0],
                                  'n_estimators': [10,50,100,200]}
```

Fig.no.B.1.4 PYTHON CODE

OUTPUTS:



```
average_performance(aucs_kNN,Accuracy,TP,TN,FP,FN)                    #print the average performance of the model
```

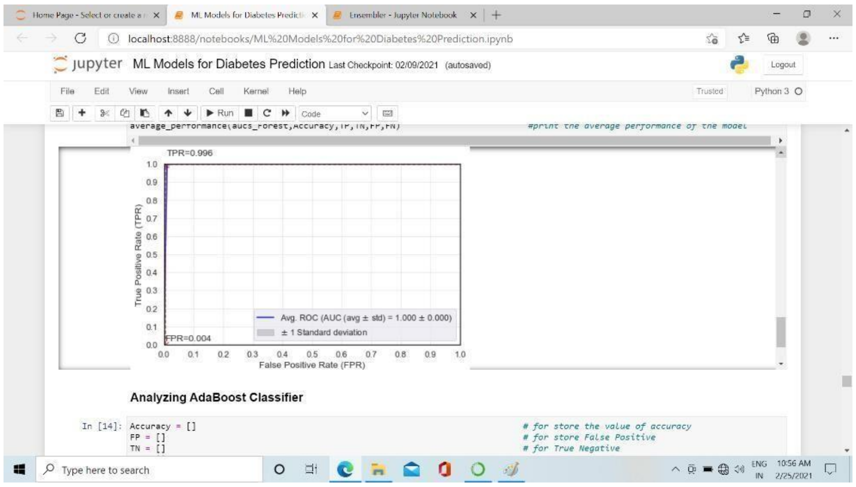
TPR=0.996
FPR=0.001
True Positive Rate (TPR)
False Positive Rate (FPR)
Avg. ROC (AUC (avg ± std) = 0.998 ± 0.004)
± 1 Standard deviation

Analyzing Decision Trees Classifier

```
In [12]: Accuracy = []                                                # for store the value of accuracy
         FP = []                                                      # for store False Positive
         TN = []                                                      # for True Negative
         FN = []                                                      # for False Negative
```

Fig.no.B.2: GRAPH FOR KNN

Here the knn gets an accuracy of 99%



```
average_performance(aucs_Forest,Accuracy,TP,TN,FP,FN)                 #print the average performance of the model
```

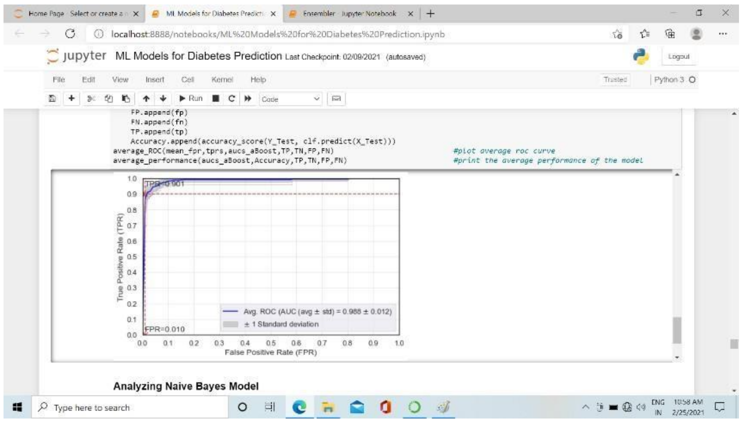
TPR=0.996
FPR=0.004
True Positive Rate (TPR)
False Positive Rate (FPR)
Avg. ROC (AUC (avg ± std) = 1.000 ± 0.000)
± 1 Standard deviation

Analyzing AdaBoost Classifier

```
In [14]: Accuracy = []                                                # for store the value of accuracy
         FP = []                                                      # for store False Positive
         TN = []                                                      # for True Negative
```

Fig.no.B.3: GRAPH OF RF

Here the rf gets an accuracy of 99%

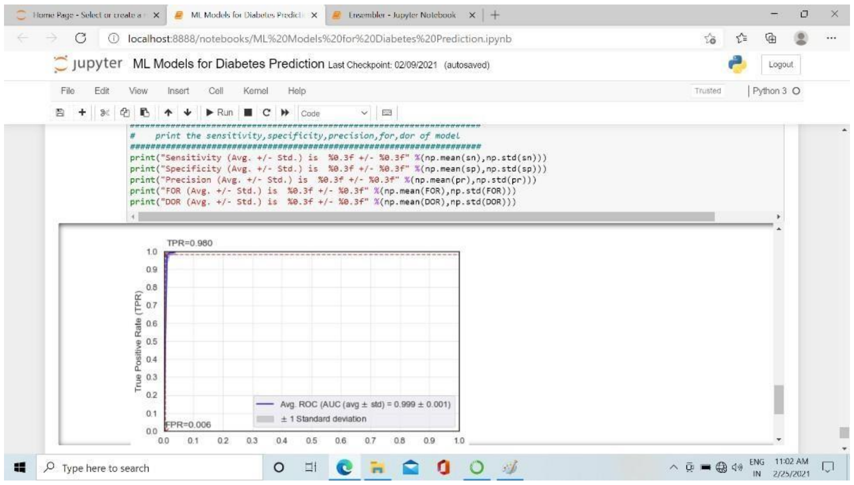


```
    FP.append(fp)
    FN.append(fn)
    TP.append(tp)
    Accuracy.append(accuracy_score(Y_Test, clf.predict(X_Test)))
average_ROC(mean_fpr,tprs,aucs_aBoost,TP,TN,FP,FN)                  #plot average roc curve
average_performance(aucs_aBoost,Accuracy,TP,TN,FP,FN)               #print the average performance of the model
```

TPR=0.901
FPR=0.010
True Positive Rate (TPR)
False Positive Rate (FPR)
Avg. ROC (AUC (avg ± std) = 0.988 ± 0.012)
± 1 Standard deviation
Analyzing Naive Bayes Model

Fig.no.B.4: GRAPH FOR ADABOOST

Here the adaboost gives an accuracy of 98%



```
#####################################################################
#    print the sensitivity,specificity,precision,for,dor of model
#####################################################################
print("Sensitivity (Avg. +/- Std.) is  %0.3f +/- %0.3f" %(np.mean(sn),np.std(sn)))
print("Specificity (Avg. +/- Std.) is  %0.3f +/- %0.3f" %(np.mean(sp),np.std(sp)))
print("Precision (Avg. +/- Std.) is  %0.3f +/- %0.3f" %(np.mean(pr),np.std(pr)))
print("FOR (Avg. +/- Std.) is  %0.3f +/- %0.3f" %(np.mean(FOR),np.std(FOR)))
print("DOR (Avg. +/- Std.) is  %0.3f +/- %0.3f" %(np.mean(DOR),np.std(DOR)))
```

Fig.no.B.5: GRAPH FOR XGBOOST

Here the xgboost gives an accuracy of 99%