MALLA REDDY UNIVERSITY

“PREDICTING DIABETES WITH MACHINELEARNING”

S. Lavan Karthik	2011CS010266
M. Vamshi Reddy	2011CS010265
P. Jayanth	2011CS010216
S. Hari Sai Ram	2011CS010264

Guided by
Mr. M. Gouthamm, Assistant Professor

SCHOOL OF ENGINEERING

COMPUTER SCIENCE & ENGINEERING

MALLA REDDY UNIVERSITY

(As per Telangana State Private Universities Act No.13 of 2020 and
G.O.Ms.No.14,Higher Education (UE) Department)

DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Project Work titled, **“PREDICTING DIABETES WITH MACHINE LEARNING”** a bonafide work of **S. Lavan Karthik (2011CS010266), M.VamshiReddy (2011CS010265), P.Jayanth (2011CS010216), HariSairam (2011CS010264)**who carried out the work under my supervision and submitted in partial fulfillment of the requirements for the award of the degree of BACHELOR TECHNOLOGY in COMPUTERSCIENCE and ENGINEERING by **Malla Reddy University** during the academic year 2022- 2023.

Guide Name: **Signature of Guide:**

Internal Examiner Name: **Signature Internal Examiner:**

External Examiner Name: **Signature External Examiner:**

HOD Signature

DECLARATION

We hereby inform that this main project entitled "**PREDICTING DIABETES WITH MACHINE LEARNING**" has been carried out and submitted in partial fulfillment for the award to the Degree of **Bachelor of Technology in Computer Science and Engineering** to **MALLA REDDY UNIVERSITY, Hyderabad** under the guidance of Mr. M. Gouthamm (M Tech), Assistant Professor, Department of COMPUTER SCIENCEAND ENGINEERING. The work embodied in this project work is original and has not been submitted in part or full for any degree of this or any degree of any other university.

Submitted By

S. Lavan Karthik	(2011CS010266)
M. Vamshi Reddy	(2011CS010265)
P. Jayanth	(2011CS010216)
S. Hari Sairam	(2011CS010264)

INDEX

LIST OF FIGURES

LIST OF TABLES

LIST OF OUTPUT SCREENS

ABSTRACT

According to the growing morbidity in recent years, in 2040, the world's diabetic patients will reach 642 million, which means that one of the ten adults in the future is suffering from diabetes. There is no doubt that this alarming figure needs great attention.With the rapid development of machine learning, machine learning has been applied to many aspects of medicalhealth. In this study, we used decision tree, random forest to predict diabetes mellitus. The results showed thatprediction with random forest could reach the highest accuracy when all the attributes were used. Type 2 diabetes occurs more commonly in middle-aged and elderly people, which is often associated with theoccurrence of obesity, hypertension, dyslipidemia, arteriosclerosis, and other diseases. Machine learning methods are widely used in predicting diabetes, and they get preferable results. Decision tree is one of popularmachine learning methods in medical field, which has grateful classification power.

Random forest generates many decision trees. So in this study, we used decision tree, random forest (RF) to predict the diabetes. Machine learning (ML) algorithms methods are used in diabetes prediction in our research. Predictive learningis a technique of machine learning in which an agent tries to build a model of its environment by trying out different actions in various circumstances. It uses knowledge of the effects its actions appear to have, turning them into planning operators. At its most basic, machine learning uses programmed algorithms that receive andanalyze input data to predict output values within an acceptable range. As new data is fed to these algorithms, they learn and optimize their operations to improve performance, developing 'intelligence' over time.

1.INTRODUCTION

1.1 INTRODUCTION OF THE PROJECT

First before we learn about diabetes predictor, let us clearly know what diabetes actually is, Diabetes, also known as diabetes mellitus, is a group of metabolic disorders characterized by a high blood sugar level over a prolonged period of time. Symptoms often include frequent urination, increased thirst and increased appetite If left untreated, diabetes can cause many health complications. Acute complications can include diabetic ketoacidosis, hyperosmolar hyperglycaemic state, or death. Serious long-term complications include cardiovascular disease, stroke, chronic kidney disease, foot ulcers, damage to the nerves, damage to the eyes, and cognitive impairment. Diabetes is due to either the pancreas not producing enough insulin, or the cells of the body not responding properly to the insulin produced. Insulin is a hormone which is responsible for helping glucose from food get into cells to be used for energy. There are three main types of diabetes mellitus. Apart from this there are mainly two types of Diabetes and let us understand them in detail.

1. Type 1 diabetes results from failure of the pancreas to produce enough insulin due to loss of beta cells. his form was previously referred to as "insulin-dependent diabetes mellitus" or "juvenile diabetes" he loss of beta cells is caused by an autoimmune response. The cause of this autoimmune response is unknown. Although Type 1 diabetes usually appears during childhood or adolescence, it can also develop in adults. Type 2 diabetes begins with insulin resistance, a condition in which cells fail to respond to insulin properly. As the disease progresses, a lack of insulin may also develop. This form was previously referred to as "non-insulin-dependent diabetes mellitus" or "adult-onset diabetes".

2. Type 2 diabetes is more common in older adults, but a significant increase in the prevalence of obesity among children has led to more cases of type 2 diabetes in younger people. The most common cause is a combination of excessive body weight and insufficient exercise.

3. Gestational diabetes is the third main form and occurs when pregnant women without a previous history of diabetes develop high blood sugar levels. In women with gestational diabetes, blood sugar usually returns to normal soon after delivery. However, women who had gestational diabetes during pregnancy have a higher risk of developing type 2 diabetes later in life.

4. Coming to our predictor it is mainly used to predict the diabetes in a person based on all the results that are obtained by various tests performed during, however our predictor does not provide any treatment or medication about the disease and it just conforms whether the person has the disease or not, this prediction is very helpful up to a great extent in this present corporate world where everything is related to money mainly.

1.1.1 OBJECTIVE

This research work aims to analyse the Diabetes dataset, design, and implement a Diabetes prediction and recommendation system utilizing machine learning classification techniques. The specific objectives of this project work are:

(i) To review existing literature along the area of diabetes diagnosis and prediction.

(ii) Design and develop a model using machine learning techniques.

(iii) To analyse the Diabetes dataset and use Support Vector Machine and Random forest algorithms to develop a prediction engine.

(iv) To identify and discuss the benefits of the designed system along with effective applications.

An effective framework for knowledge required Machine learning has been used successfully by researchers for the prognosis and/or diagnosis of diabetes for active and accurate decision making. Therefore, this project focuses on the use of machine learning techniques on a set of data collected which is an online dataset to uncover hidden patterns and predict diabetes based on the dataset collected. Support Vector Machine and random forest are proposed for use in the prediction of diabetes in a patient to ensure that the information gotten from the system built based on these techniques are reliable. Good demonstration of a real-world machine learning process. As a clinician I have concerns about using this dataset without some medical expertise. For example, you can't have triceps thickness or insulin levels of zero. This means the test was not done. Is imputation legitimate in this situation? A pregnancy level of zero could mean it was not asked or it could mean no pregnancies. We don't know which. The column on pedigree is complicated and probably should be deleted and not used.

Predicting Diabetes Mellitus in Patients Using Machine Learning. Diabetes Mellitus, the disease caused by hyperglycemia and can be classified as type 1 and type 2.

1. Type 1 diabetes is often discovered in early diagnosis in young children and adolescents. However, given recent evidence, it is shown that the condition can develop at any age and therefore people of all ages are susceptible to development.
2. Type 2 diabetes is an autoimmune disease that causes the insulin producing beta cells in the pancreas to be eradicated thereby causing prevention for the body to produce adequate levels of insulin to regulate blood glucose levels.

Another terminology for type 1 diabetes is Insulin Dependent Diabetes. On the contrary, type 2 diabetes, which is the most common diagnosis in the general population, is a metabolic disorder resulting in hyperglycemia. This is often the prognosis of the body's inability to utilize the insulin it produces or ability to produce insulin. Gestational diabetes appears in pregnant women who suddenly develop high blood sugar. In two thirds of the cases, it will reappear during subsequent pregnancies. There is a great chance that type 1 or type 2 diabetes will occur after a pregnancy affected by gestational diabetes.

Symptoms of Diabetes

1. Frequent Urination
2. Increased thirst
3. Tired/Sleepiness
4. Weight loss
5. Blurred vision
6. Mood swings
7. Confusion and difficulty concentrating
8. frequent infection

1.1.2 METHODOLOGY

In this section we shall learn about the various classifiers used in machine learning to predict diabetes. We shall also explain our proposed methodology to improve the accuracy. Five different methods were used in this paper. The different methods used are defined below. The output is the accuracy metrics of the machine learning models. Then, the model can be used in prediction.

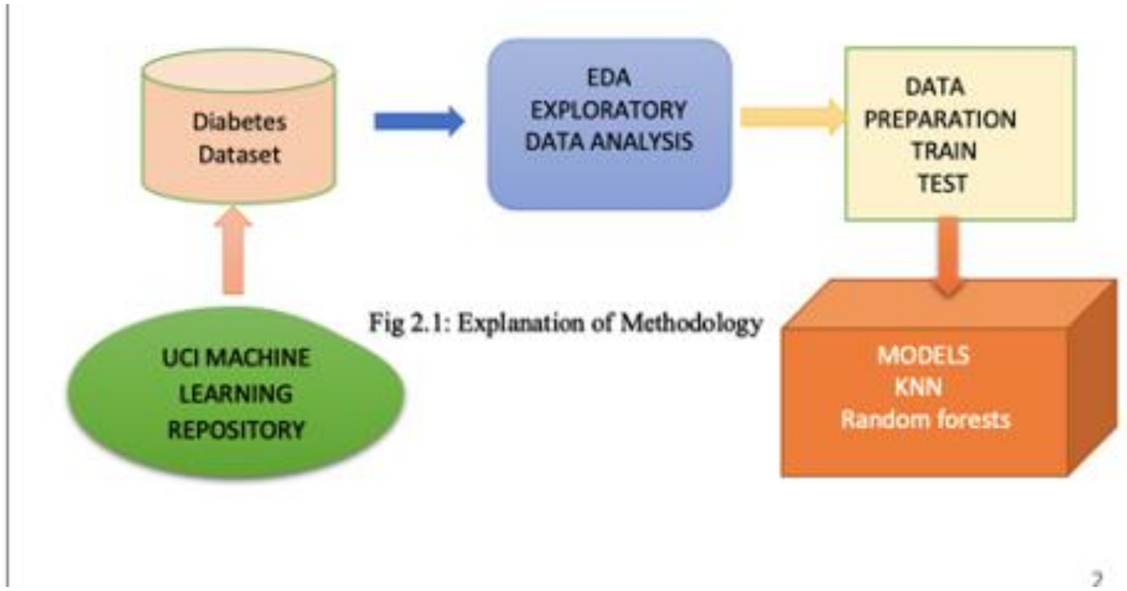

Diabetes Dataset
EDA EXPLORATORY DATA ANALYSIS
DATA PREPARATION TRAIN TEST
UCI MACHINE LEARNING REPOSITORY
Fig 2.1: Explanation of Methodology
MODELS KNN Random forests
2

Fig 1

1.1.3 ALGORITHM

STEP1: Gathering the data from the Kaggle and understand about the dataset and preparing the data by removing the missing values.

STEP 2: Now we split the gathered data into training and testing dataset.

STEP 3: Now using training data we create a Random Forest Classification model.

STEP 4: Using the testing data we test the created Random Forest Classification model and import the pickle file.

STEP 5: Using the model now we predict diabetes.

1.2 EXISTING SYSTEM

Existing many research handled for the diabetes detection. Data mining approach like clustering, classification was studied in existing system. Diabetes prediction using algorithm such as K-NN, K-Means, branch and bound algorithm was proposed. A basic diabetic dataset is chosen for carrying out the comparative analysis. The importance of feature analysis for predicting diabetes by employing machine learning technique is discussed.

1.3 ISSUES IN EXISTING SYSTEM

High false positives. There is no interactive tool for users to predict diabetes.

1. Costing: The Existing system is high cost and this is main reason most of the system is failed.
2. Technology Complexity: Most of system is complex to understand, not user friendly as compare to our proposed system.
3. Time Consuming Feature: In existing system, the performance is low and most of the time system gets hanged due to load.
4. Not Easy to Understand: Systems re complex to understand and they were not user friendly.

1.4 PROPOSED SYSTEM

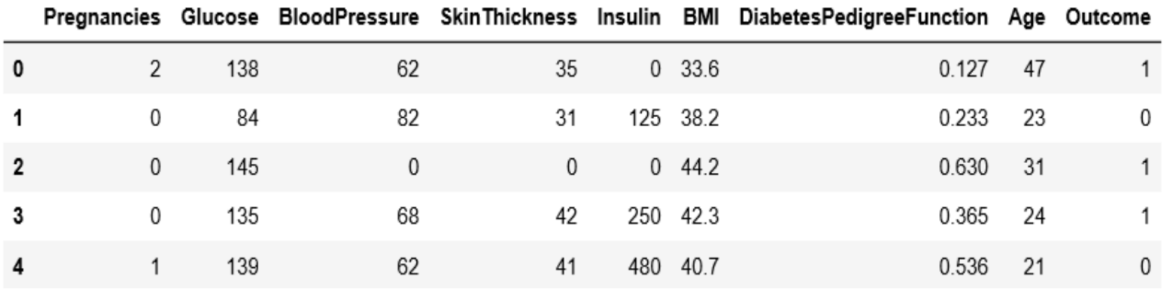
1. The Proposed System study is classification of Indian PIMA dataset for diabetes as a binary classification problem.
2. This is proposed to achieve through machine learning classification algorithms and Regression.
3. For machine learning SVM algorithm, K-NN algorithm, Random Forest algorithm and Logistic Regression are proposed.
4. The proposed system improves accuracy of prediction through Random Forest Classification.
5. The proposed is also has the frontend with the angular code. The website is fully dynamic and the we can user friendly website.
6. Then the project has deployed in the AWS EC2 backend is deployed and the front is deploy in the s3 bucket.

1.4.1 ADVANTAGES OF PROPOSED SYSTEM

1. Interactive application, in which user can give a single input to arrive the prediction.
2. Accuracy is improved using machine learning techniques.
3. By this project we can know about the machine learning algorithms and AWS basics.

1.5 DATASET DESCRIPTION

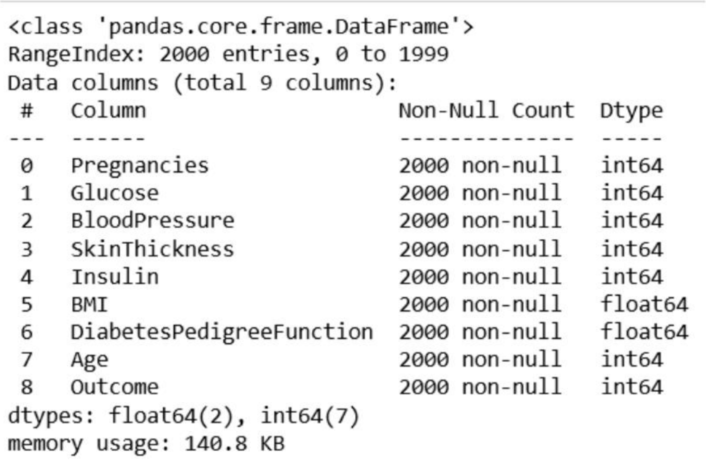
The diabetes data set was originated from https://www.kaggle.com/johndasilva/diabetes. Diabetes dataset containing 2000 cases. The objective is to predict based on the measures to predict if the patient is diabetic or not.



	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	2	138	62	35	0	33.6	0.127	47	1
1	0	84	82	31	125	38.2	0.233	23	0
2	0	145	0	0	0	44.2	0.630	31	1
3	0	135	68	42	250	42.3	0.365	24	1
4	1	139	62	41	480	40.7	0.536	21	0

Table 1

1. The diabetes data set consists of 2000 data points, with 9 features each.
2. “Outcome” is the feature we are going to predict, 0 means No diabetes, 1 means diabetes.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               2000 non-null   int64  
 1   Glucose                   2000 non-null   int64  
 2   BloodPressure             2000 non-null   int64  
 3   SkinThickness             2000 non-null   int64  
 4   Insulin                   2000 non-null   int64  
 5   BMI                       2000 non-null   float64
 6   DiabetesPedigreeFunction  2000 non-null   float64
 7   Age                       2000 non-null   int64  
 8   Outcome                   2000 non-null   int64  
dtypes: float64(2), int64(7)
memory usage: 140.8 KB
```

There are no null values in dataset.

1.6 DATASET CONTAINS

Serial no	Attributes names	description
1	Pregnancies	Number of times of pregnant.
2	Glucose	Plasma glucose concentration.
3	Blood Pressure	Diastolic blood pressure.
4	Skin Thickness	Triceps skin fold thickness(mm).
5	Insulin	2-h serum insulin.
6	BMI	Body mass index.
7	Diabetes pedigree function	Diabetes pedigree function
8	Age	Age of patient.
9	Outcome	Class variable(0 or 1).

Table 2

2.ANALYSIS

2.1 REQUIREMENTS ANALYSIS

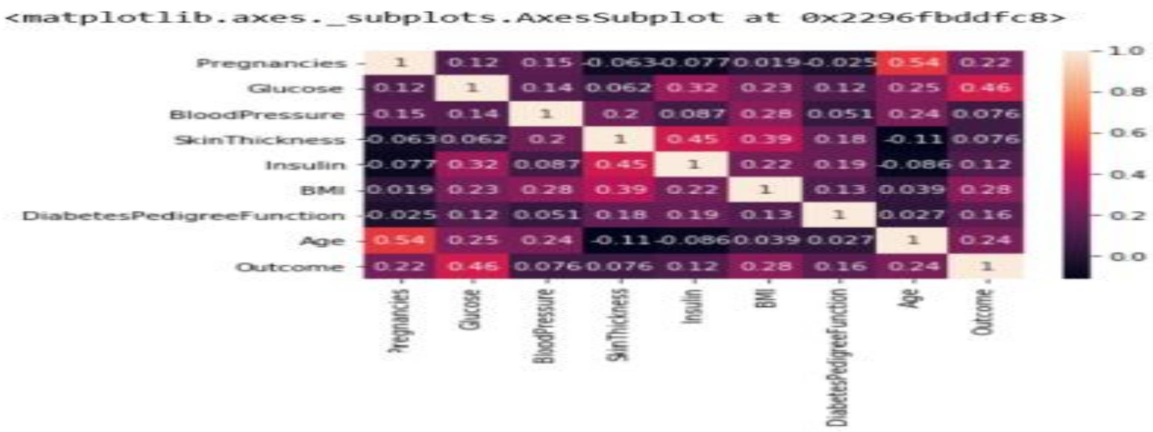
The requirement analysis of the entire tool, features, software, hardware, and logistics were untaken at this step. The major feature of the system dells on the benefit of automatizing the process of the information given to the patients for their prevention. This will ensure not the information also can know about their body status and the fitness. The reliability is more important in this analysis.

2.1.1 FUNCTIONAL REQUIREMENT ANALYSIS

The programming level is used in the website is only purely depend on the frontend if we want to add the database, we will use the backend but for now the website no need of the backendfor giving the information we need only the frontend and look and feel and information we give tothe user is more enough for the cultivation of the crop form the cultivation of land to yielding of the crop. The website contains the details of the patient and as well as the person can get the information the health organization. It is easy to see that there is no single feature that has a very high correlation with our outcome value. Some features have a negative correlation with the outcome value and some have positive.

2.1.1.1 Graphs of the model

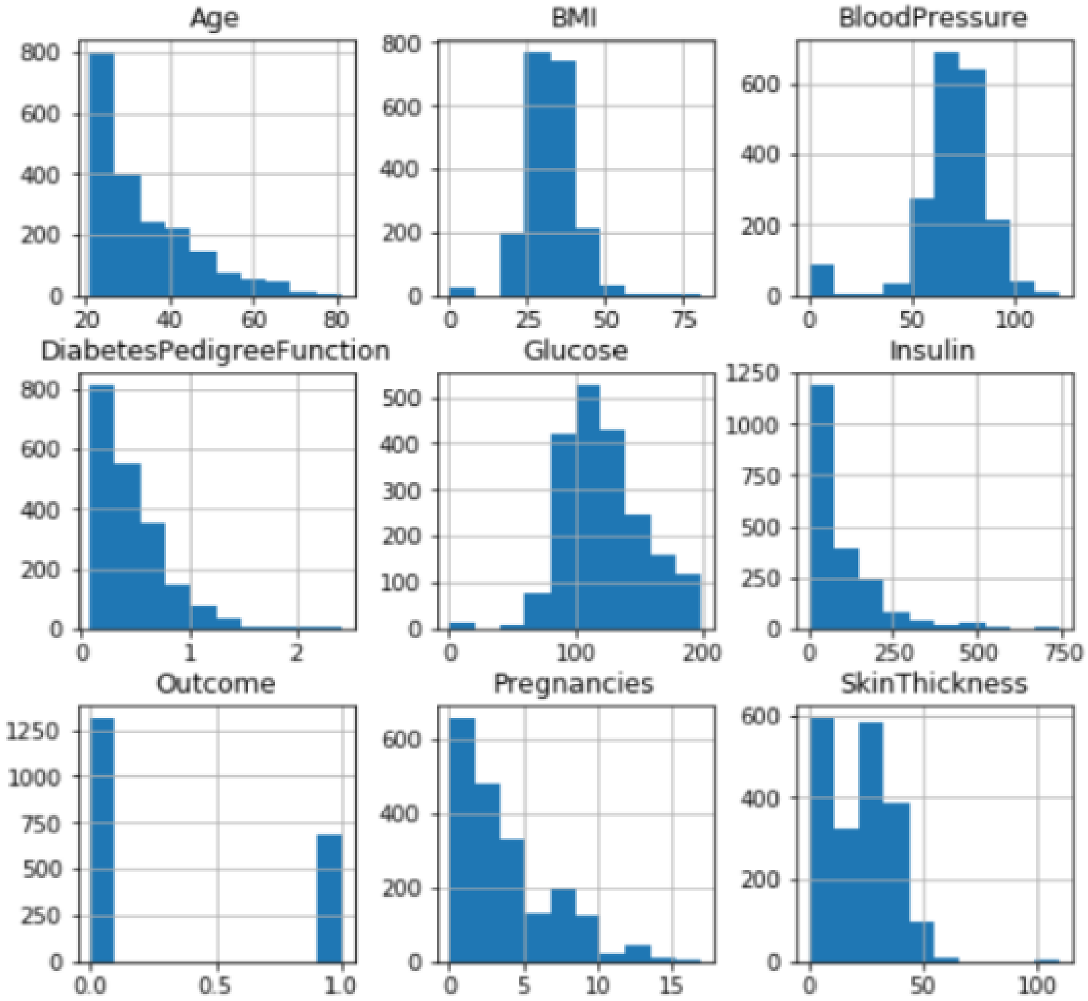
Confusion matrix



```
<matplotlib.axes._subplots.AxesSubplot at 0x2296fbddfc8>
```

Graph 1

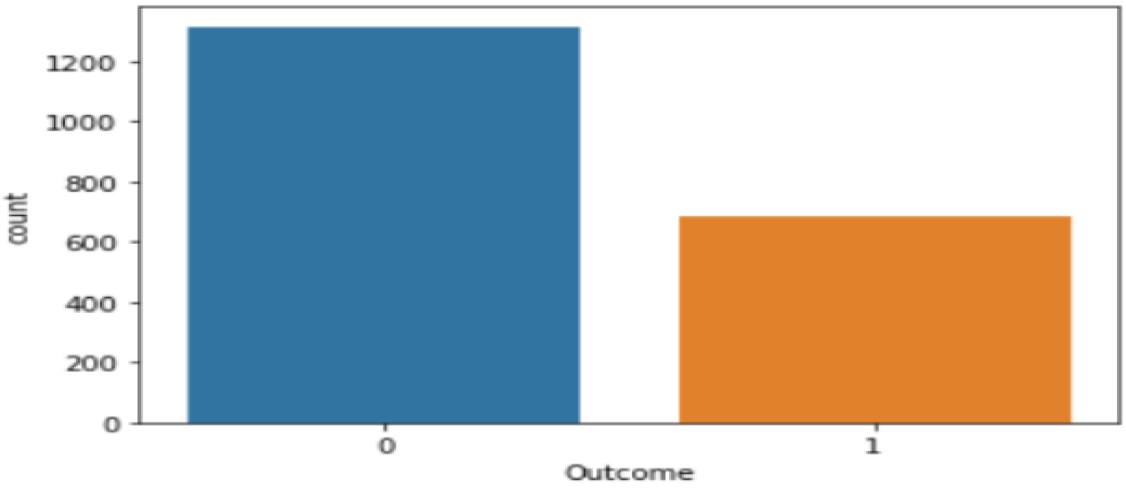
2.1.1.2 Histogram



Graph 2

Let’s take a look at the plots. It shows how each feature and label is distributed along different ranges, which further confirms the need for scaling. Next, wherever you see discrete bars, it basically, means that each of these is actually a categorical variable. We will need to handle these categorical variables before applying Machine Learning. Our outcome labels have two classes, 0 for no disease and 1 for disease.

2.1.1.3 Box plot



Graph 3

The above graph shows that the data is biased towards datapoints having outcome value as 0 where it means that diabetes was not present actually. The number of non-diabetics is almost twice the number of diabetic patients.

2.1.2 USER AND SYSTEM REQUIREMENTS

User requirements are the only login with the name and email or phone number and go through the and collect the information from the website. User requirements also include the user suggestions will be taken and they can encourage theother users also by giving the information about their crop yielding. System requirements are minimal which is a basic computer/ phone with access to the interneton any operating system

2.1.3 NON-FUNCTIONAL REQUIREMENTS

1. Speed: The speedy responsive website is essential for customer satisfaction and to retain the customer on the platform.
2. Portability: The application must be portable and must not require a lot of resources from the users' devices.
3. Compatibility: The web application is compatible with all the major operating systems with an internet connection and with any modern browser.
4. Information: The user is satisfactory with the given information or not is the main thing in the website.
5. Security: The given data of the user is secure are not no one can access the data of the user.

2.2 MODULES DESCRIPTION

We have only the single page application to know that the diabetes is present or not for the patients.

USER

The User can register the first. While registering he required a valid User email and mobile for further communications. Once the User registers, then the admin can activate the User. Once the admin activates the User then the User can login into our system. After login User will add the data to predict diabetes prediction is present or not.

ADMIN

Admin can login with his credentials. Once he logs in he can activate the users. The activated users only login in our applications. The admin will store csv data into our database. we can implement Random Forest classification algorithm to predict diabetes and also we can perform cross validation.

MACHINE LEARNING

Machine learning refers to the computer's acquisition of a kind of ability to make predictive judgments and make the best decisions by analyzing and learning a large number of existing data. The representation algorithms include deep learning, artificial neural networks, decision trees, enhancement algorithms and so on. The key way for computers to acquire artificial intelligence is machine learning. Nowadays, machine learning plays an important role in various fields of artificial intelligence. Whether in aspects of internet search, biometric identification, auto driving, Mars robot, or in American presidential election, military decision assistants and so on, basically, as long as there is a need for data analysis, machine learning can be used to play a role.

2.2.1 FLOW OF ALGORITHM:

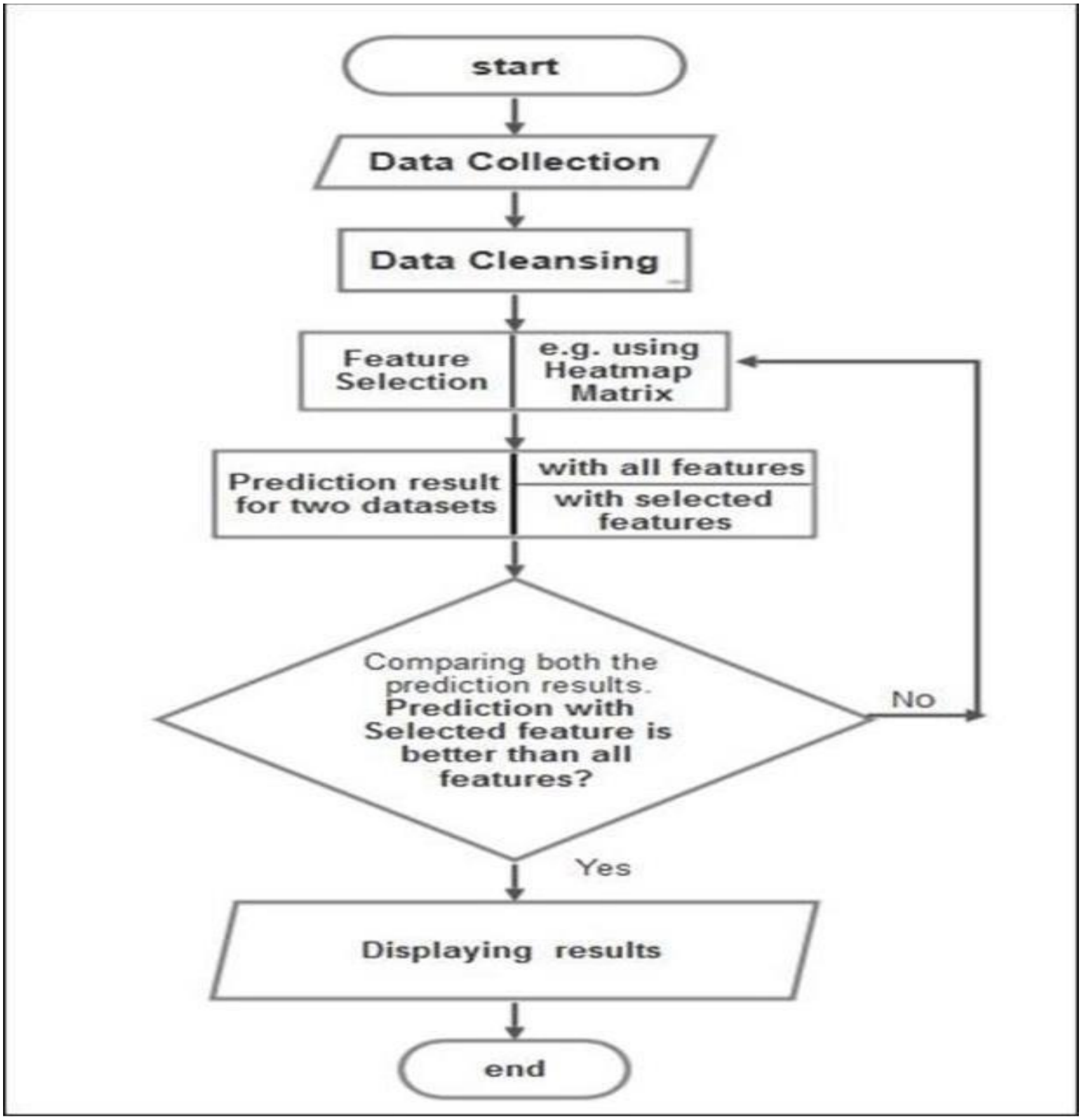

start
Data Collection
Data Cleansing
Feature Selection
e.g. using Heatmap Matrix
Prediction result for two datasets
with all features
with selected features
Comparing both the prediction results.
Prediction with Selected feature is better than all features?
No
Yes
Displaying results
end

Fig 1.2

2.3 FEASIBILITY STUDY

A feasibility study is an assessment of the practicality of a proposed plan or project. A feasibility study analyzes the viability of a project to determine whether the project or venture is likely to succeed.

2.3.1 TECHNICAL FEASIBILITY

Technical feasibility inspects whether software can be built at all with available tools and experts.The proposed system will be built using the following technologies:

1. HTML, CSS, Angular for front-end framework.
2. Machine learning code and FastAPI for the back-end runtime environment.
3. AWS for deploying of the frontend in S3 bucket and backend code in EC2 instance.

The prototype is largely built on the Machine Learning and allows for dynamic and easy updates of changes in the prototype.

2.3.2 OPERATIONAL FEASIBILITY

Operational feasibility is the measure of how well a proposed system solves the problems and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. Operational Feasibility reviews the willingness of the organization to support the proposed system. This is probability the most difficult of the feasibilities to gauge. In order to determine this feasibility, It is important to understand the management commitment to the proposed project. If the request was initiated by management, it is likely that there is management support and the system will be accepted and used.

2.3.3 BEHAVIORAL FEASIBILITY

The prototype will evaluate and estimates the user attitude or behavior towards the development of a new system. It helps in determining if the system requires special effort to educate the user or not. The prototype is proposed with the behavioral feasibility in consideration.

2.4 PROCESS MODEL USED

MACHINE LEARNING MODELS USED IN THE PROJECT

K-Nearest Neighbors

The K-NN algorithm is arguably the simplest machine learning algorithm. Building the model consists only of storing the training data set. To make a prediction for a new data point, the algorithm finds the closest data points in the training data set, its “nearest neighbors.”

First, let’s investigate whether we can confirm the connection between model complexity and accuracy:

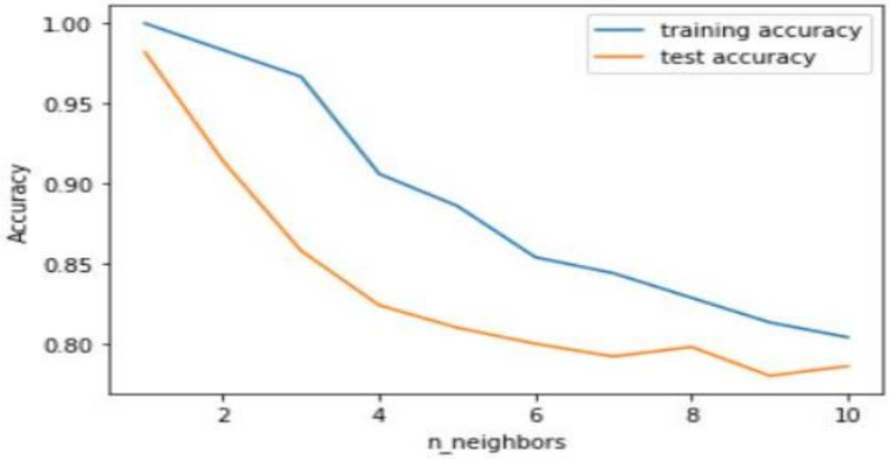


Fig1.3

The above plot shows the training and test set accuracy on the y-axis against the setting of n-neighbors on the x-axis. Considering if we choose one single nearest neighbor, the prediction on the training set is perfect. But when more neighbors are considered, the training accuracy drops, indicating that using the single nearest neighbor leads to a model that is too complex. The best performance is somewhere around 9 neighbors.

Training Accuracy	0.81
Testing Accuracy	0.78

Decision Tree

This classifier creates a decision tree based on which, it assigns the class values to each data point. Here, we can vary the maximum number of features to be considered while creating the model.

Feature Importance in Decision Trees

Feature importance rates how important each feature is for the decision a tree makes. It is a number between 0 and 1 for each feature, where 0 means "not used at all" and 1 means "perfectly predicts the target".

Training Accuracy	0.78
Testing Accuracy	0.75

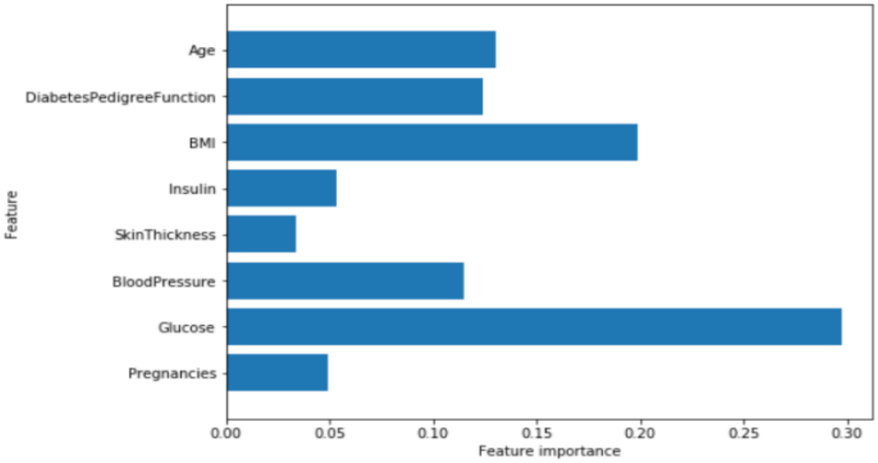


Fig 1.4

Feature "Glucose" is by far the most important feature.

Random Forest Classification

This classifier takes the concept of decision trees to the next level. It creates a forest of trees where each tree is formed by a random selection of features from the total features.We divide the data into training and testing. Then we use the Random Forest Classifier to classify the patient is having diabetes or not.This classifier takes the concept of decision trees to the next level. It creates a forest of trees where each tree is formed by a random selection of features from the total features.

Feature importance in Random Forest

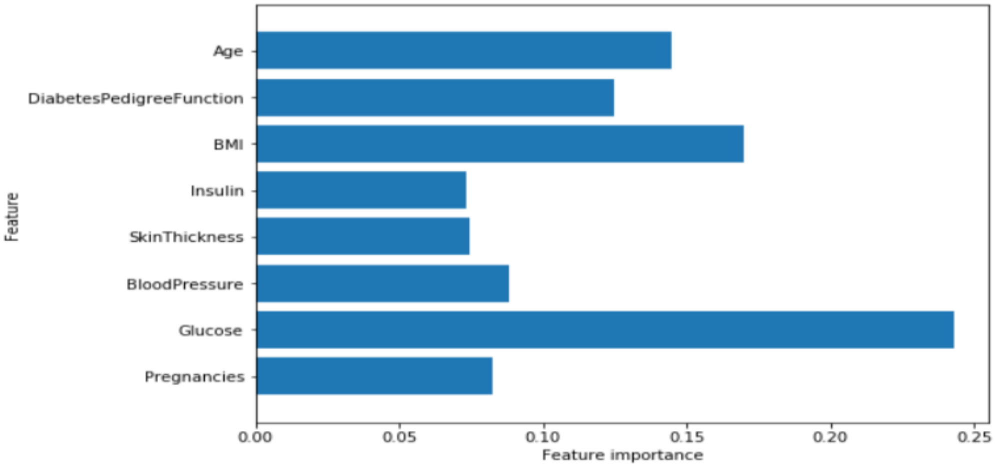


Fig 1.5

Similarly, to the single decision tree, the random forest also gives a lot of importance to the "Glucose" feature, but it also chooses "BMI" to be the 2nd most informative feature overall.

Support Vector Machine

This classifier aims at forming a hyper plane that can separate the classes as much as possible by adjusting the distance between the data points and the hyper plane. There are several kernels based on which the hyper plane is decided. I tried four kernels namely, linear, poly, rbf and sigmoid.

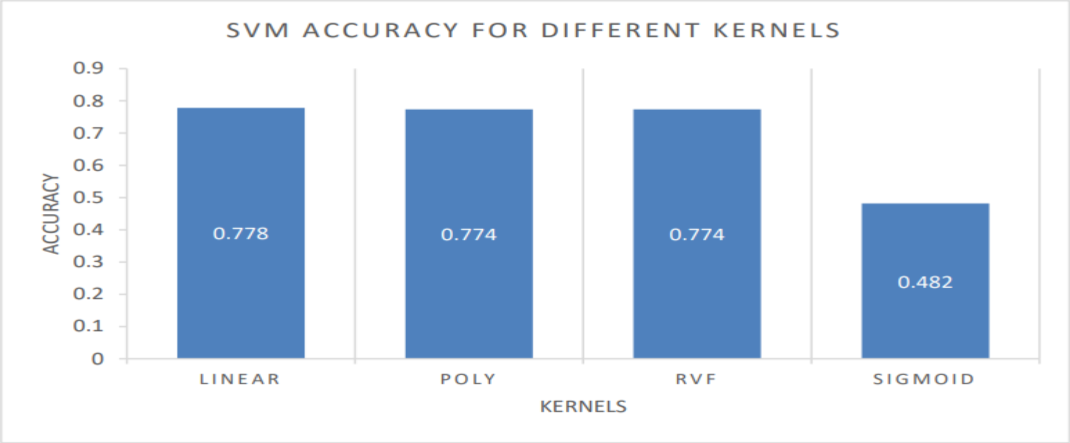


Fig 1.6

As can be seen from the plot above, the linear kernel performed the best for this dataset and achieved a score of 77%.

Accuracy Comparison

Models	Training accuracy	Testing accuracy
Decision Tree	78%	78%
k-Nearest Neighbors	81%	78%
Random Forest	94%	99%
SVM	76%	77%

There is more accuracy for the random forest classification, so we use the random forest classification to create the pickle file.

2.5 DIAGRAMS

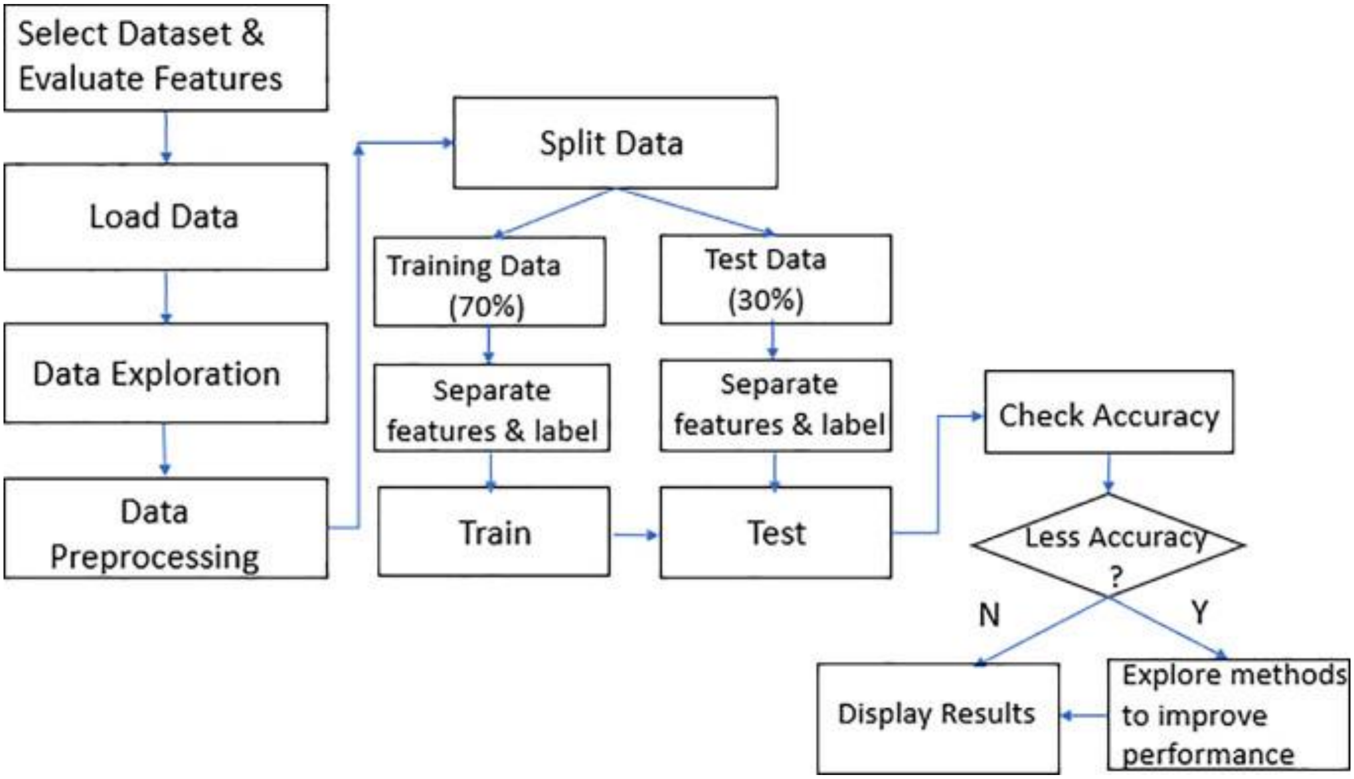


Fig 1.7: process model

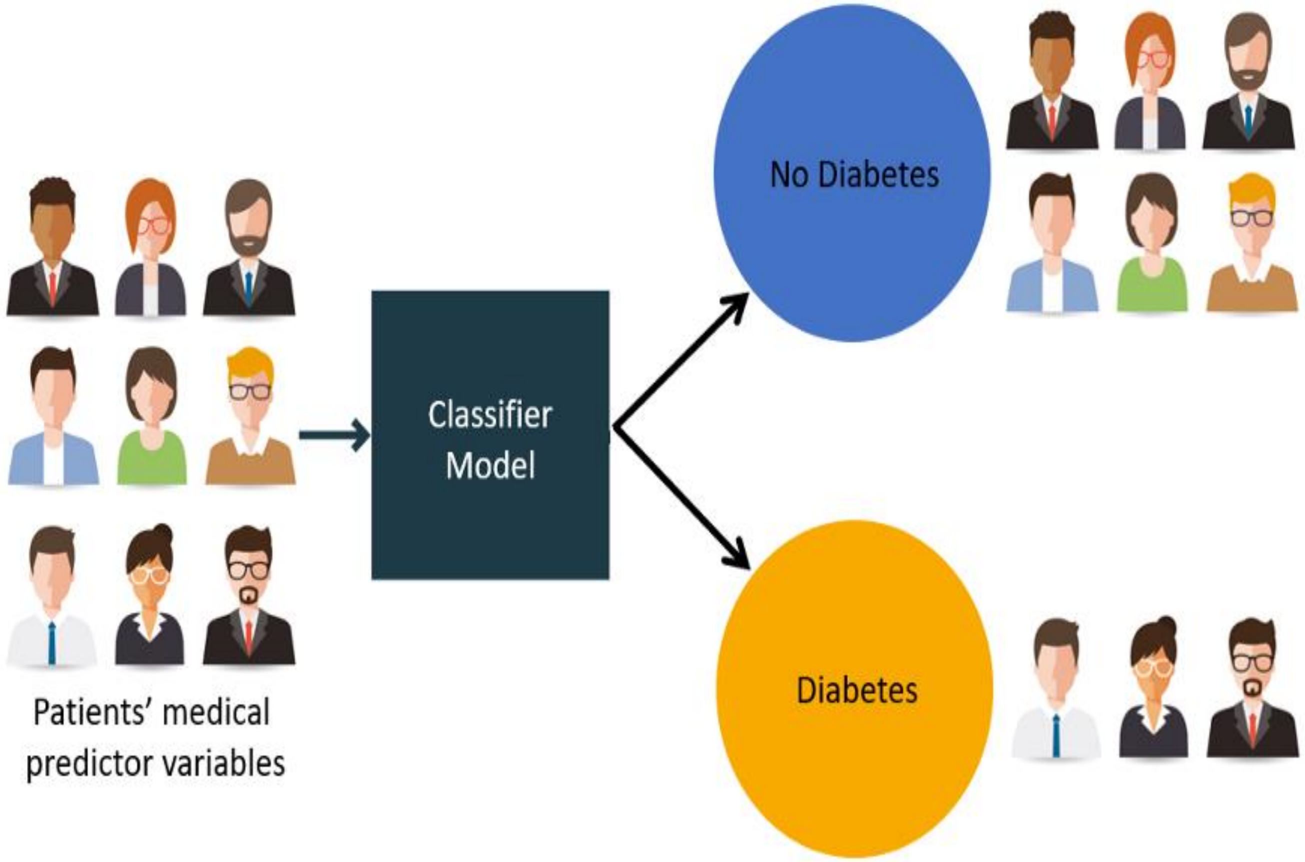


Fig 2

REQUIREMENTS

SOFTWARE REQUIREMENTS

1. Python(backend)
2. FastAPI(backend)
3. Uvicorn(backend)
4. Angular(frontend)
5. Node Js(frontend)

HARDWARE REQUIREMENTS

1. Processor: minimum 1 GHz or more.
2. Ethernet connection (LAN) OR a wireless adapter (Wi-Fi).
3. Hard drive: Minimum 32 GB;Recommended 64 GB.
4. Memory (RAM): Minimum 1 GB; Recommended 4 GB.

3.DESIGN

3.1 INTRODUCTION TO DESIGN

The design is to be created in the angular environment for the dynamic website we used the angularfor the best website and the look and feel. If you want an amazing agriculture website that stands out from the competition, work with professional designers from Fire art. Hire designers to make your vision come to life or host a design contest and get ideas from designers around the world. Agreat website shows the world who you are, makes people remember you, and helps potential customers understand if they found what they were looking for. Websites communicate all of that through color, shape and other design elements. Learn how to make your agriculture website tell your brand’s story. you are looking for visual design concepts and inspirations to develop the next agriculture website or app, you might be at the right place. Have a look at our range of the most effective agriculture web site design concepts below to discover a website look that works for you. We use the angular for the frontend. To represent the outcome for the machine learning we use the angular application. In the angular we create the service component and we connect the machine learning model to the service by deploying the model in the AWS EC2 instance and that IP address is to be call in the service component in the angular.

After completing the connection, we have to run the angular project. To check the project then after You have run the IP address then the angular project should run then after the machine learning model Should run in the instances also; we can deploy the angular application in the instance as well as in theS3 bucket of the AWS management console. We see in the instances as well as in the S3 bucket also.

Deploying in the instances: IP address should run with the port at which it has running.

3.2DFD/UML DIAGRAM

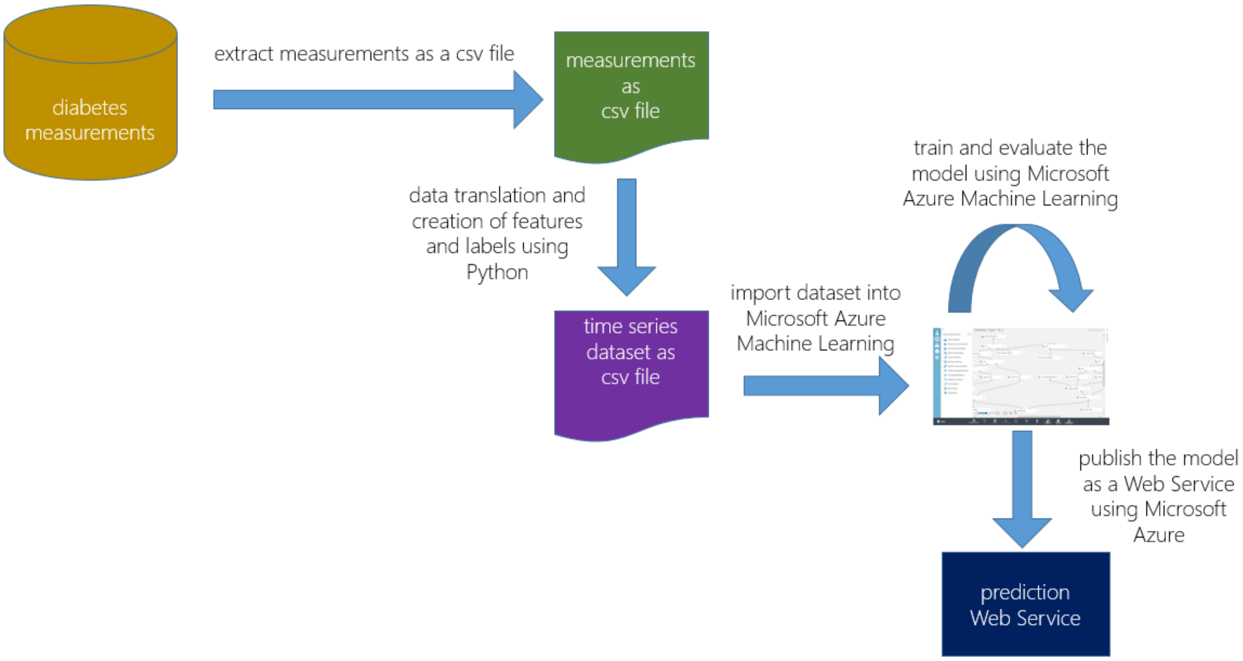

diabetes measurements
extract measurements as a csv file
measurements as csv file
data translation and creation of features and labels using Python
time series dataset as csv file
import dataset into Microsoft Azure Machine Learning
train and evaluate the model using Microsoft Azure Machine Learning
publish the model as a Web Service using Microsoft Azure
prediction Web Service

Fig 3: uml diagram

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

3.2.1 SEQUENCE DIAGRAM

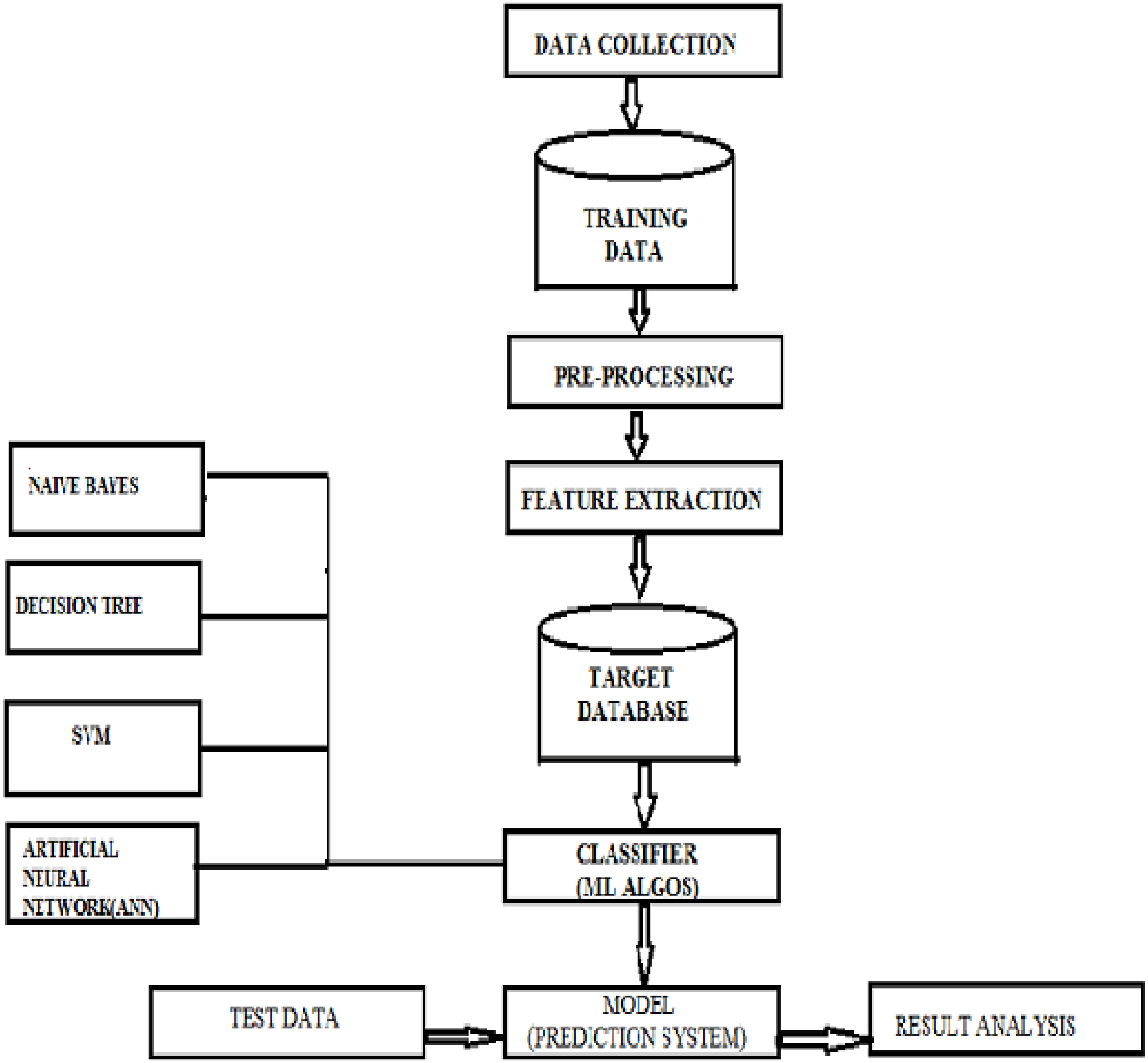


Fig 3.1: sequence diagram

Sequence Diagram represents the sequence of steps as follows:

1. User will submit review.
2. The review gets analyzed by the analyzer.
3. After analysis, the result is displayed to the user whether review is positive or negative

3.2.2 USECASE DIAGRAM

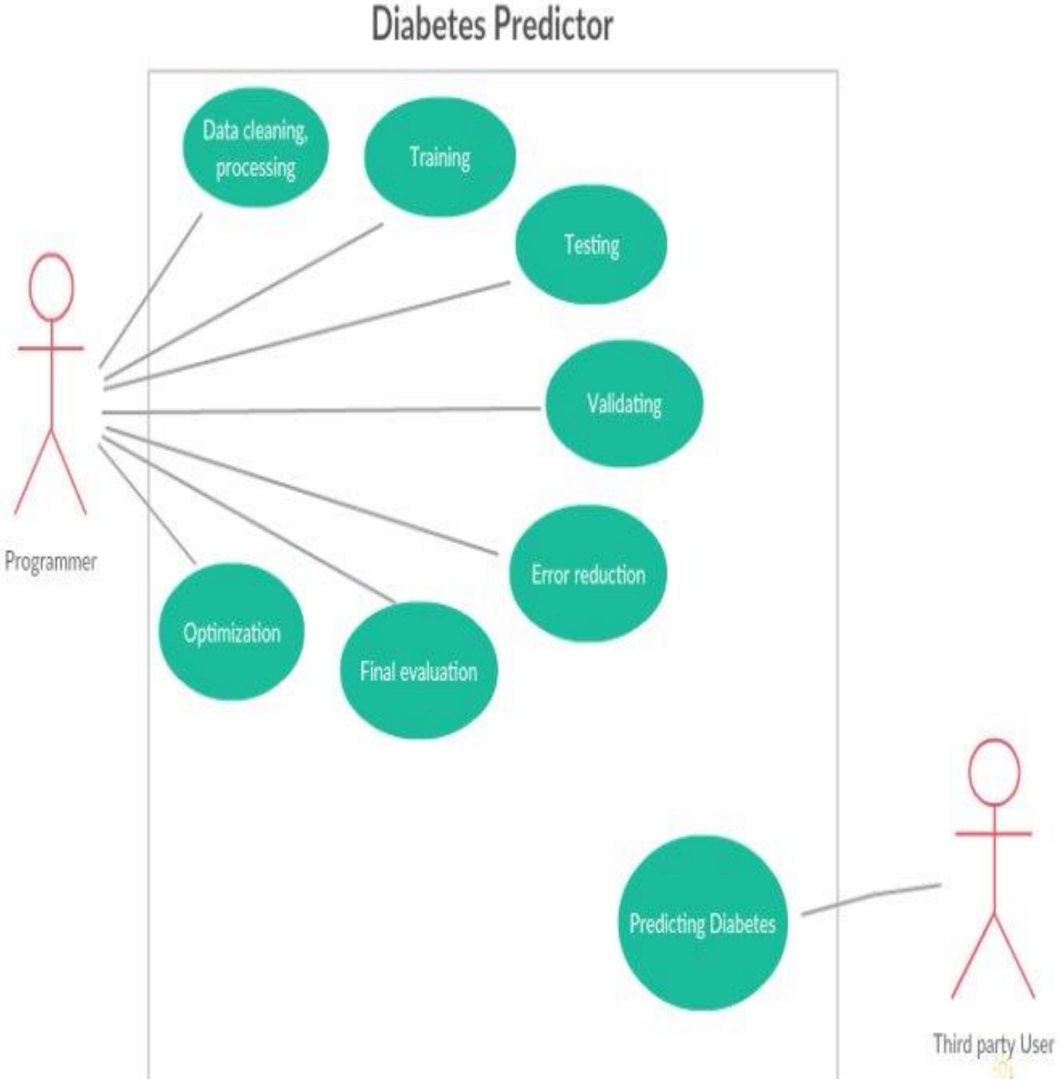


Fig 3.2: use case diagram

UML UseCase Diagram Tutorial the UML Class diagram is a graphical notation used to construct and visualize object oriented systems. A class diagram in the Unified ModellingLanguage (UML) is a type of static structure diagram that describes the structure of a system by showing the systems.

3.2.3 ACTIVITY DIAGRAM

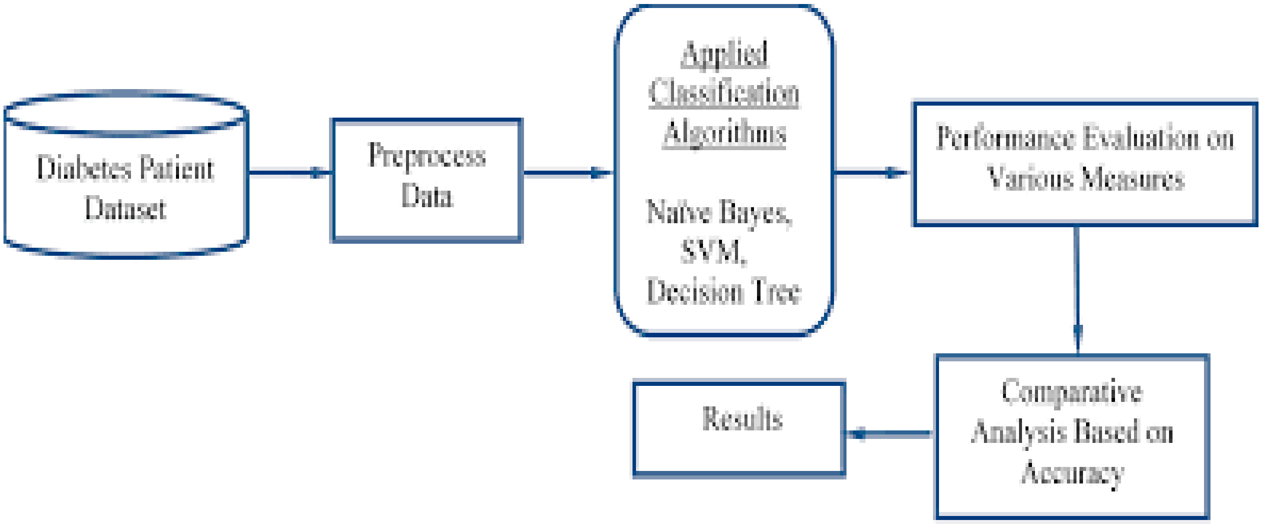

Diabetes Patient Dataset
Preprocess Data
Applied Classification Algorithms
Naïve Bayes, SVM, Decision Tree
Performance Evaluation on Various Measures
Comparative Analysis Based on Accuracy
Results

Fig 3.3: activity diagram

Activity diagram for online shopping system The activity diagram used to describe flowof activity through a series of actions. Activity diagram is an important diagram to describe the system. The activity described as a action or operation of the system.

4.IMPLEMENTATION OF CODING

4.1 ADD ALGORITHM THEORY AND ALGORITHM

An Algorithm is a procedure to solve a particular problem in a finite number of steps for a finite- sized input. The algorithms can be classified in various ways. The first thing we'll have to do is agree on what an algorithm is. Simply put, an algorithm is a recipe used to solve a problem. From a human's point of view, an example of an algorithm could be howto get up in the morning. Although it might sound simple, it can get very complicated, very quickly.Computers are machines and machines don't think. Therefore, we have to describe all of the algorithm's steps precisely. With this, we get to the first property of an algorithm **- it must be elementary** (consist of the final number of simple and comprehensible steps, commands). "Get outof bed" certainly isn't an algorithm. "Open your eyes, take the blanket off, sit up, put your legs on the ground and stand up" - this one sound concrete enough to be a true algorithm. However, we'regoing to keep things relevant to IT, so we'll solve tasks like sorting items by their size or searchingelements by their values. Sorting and searching are two basic tasks which computers do all the time,so they have to be thought out and optimized very well. Otherwise, they'd take up too much time that could be used for other processes.

Random Forest Classification

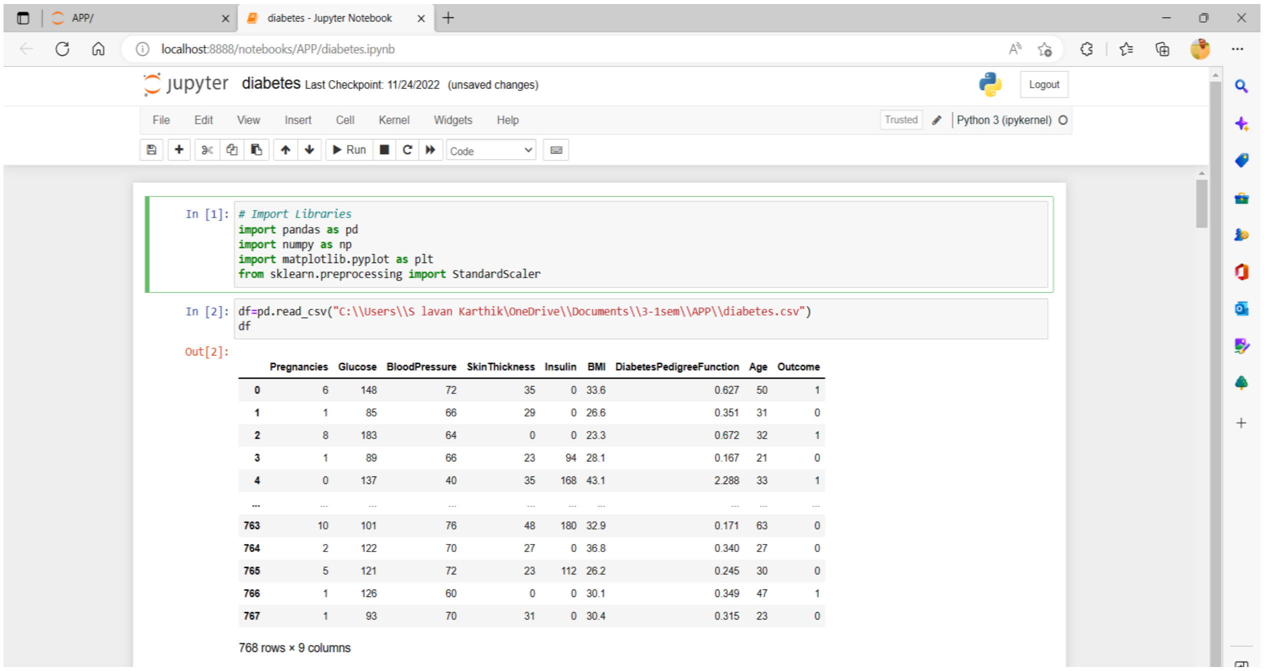
This classifier takes the concept of decision trees to the next level. It creates a forest of trees where each tree is formed by a random selection of features from the total features. We divide the data into training and testing. Then we use the Random Forest Classifier to classify the patient is having diabetes or not. This classifier takes the concept of decision trees to the next level. It creates a forest of trees where each tree is formed by a random selection of features from the total features.

Training Accuracy	1.00
Testing Accuracy	0.975

4.2 CODE REQUIRED

The project is created in the angular for the more dynamic process. The code is given below for the execution of the project. Create angular project and the place the code in the following methods. The algorithm used in the model is RANDOM FOREST CLASSIFICATION the Jupiter notebook is used in the pickle file is created and the application is deploy in the cloud.

The backend code for the machine learning:

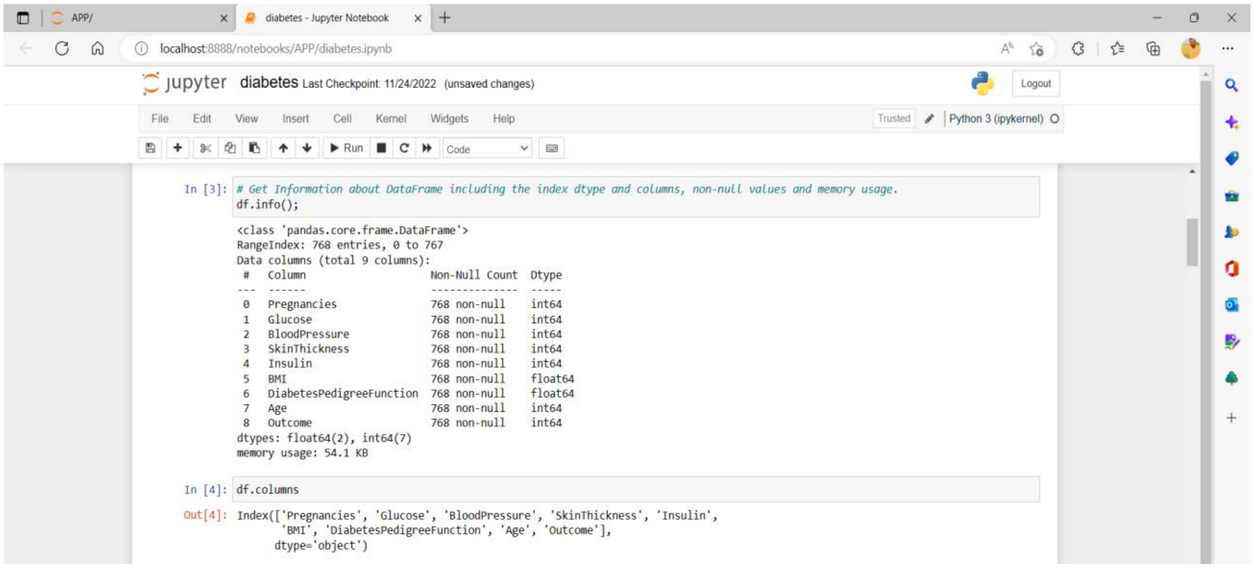

APP/
diabetes - Jupyter Notebook
localhost:8888/notebooks/APP/diabetes.ipynb
jupyter diabetes Last Checkpoint: 11/24/2022 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [1]: # Import Libraries
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        from sklearn.preprocessing import StandardScaler

In [2]: df=pd.read_csv("C:\\Users\\S lavan Karthik\OneDrive\\Documents\\3-1sem\\APP\\diabetes.csv")
        df

Out[2]:
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

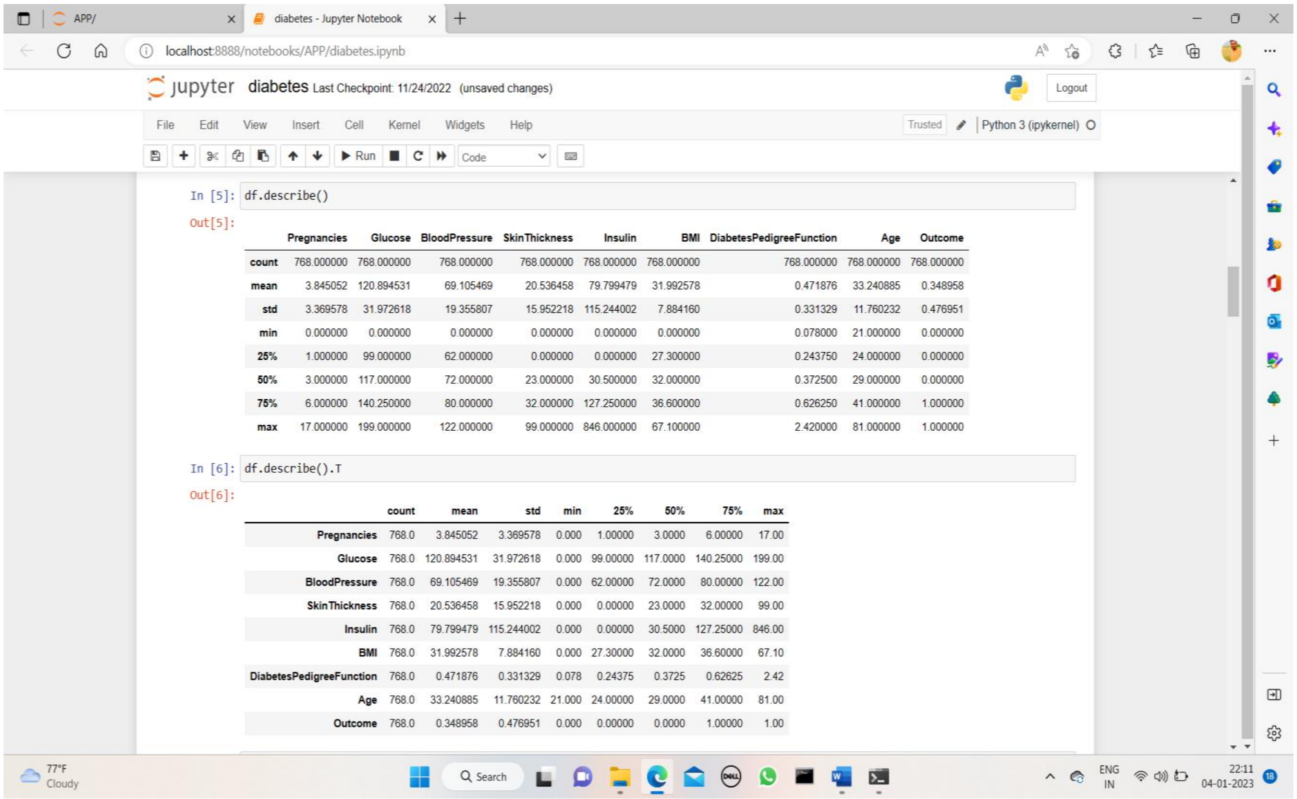
768 rows × 9 columns

Fig 4.1: importing of libraries and loading dataset.


APP/
diabetes - Jupyter Notebook
localhost:8888/notebooks/APP/diabetes.ipynb
jupyter diabetes Last Checkpoint: 11/24/2022 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [3]: # Get Information about DataFrame including the index dtype and columns, non-null values and memory usage.
        df.info();

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 768 entries, 0 to 767
        Data columns (total 9 columns):
         #   Column                    Non-Null Count  Dtype  
        ---  ------                    --------------  -----  
         0   Pregnancies               768 non-null    int64  
         1   Glucose                   768 non-null    int64  
         2   BloodPressure             768 non-null    int64  
         3   SkinThickness             768 non-null    int64  
         4   Insulin                   768 non-null    int64  
         5   BMI                       768 non-null    float64
         6   DiabetesPedigreeFunction  768 non-null    float64
         7   Age                       768 non-null    int64  
         8   Outcome                   768 non-null    int64  
        dtypes: float64(2), int64(7)
        memory usage: 54.1 KB

In [4]: df.columns

Out[4]: Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
               'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
              dtype='object')
```

Fig 4.2: EDA


APP/
diabetes - Jupyter Notebook
localhost:8888/notebooks/APP/diabetes.ipynb
jupyter diabetes Last Checkpoint: 11/24/2022 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
77°F
Cloudy
Search
ENG
IN
22:11
04-01-2023

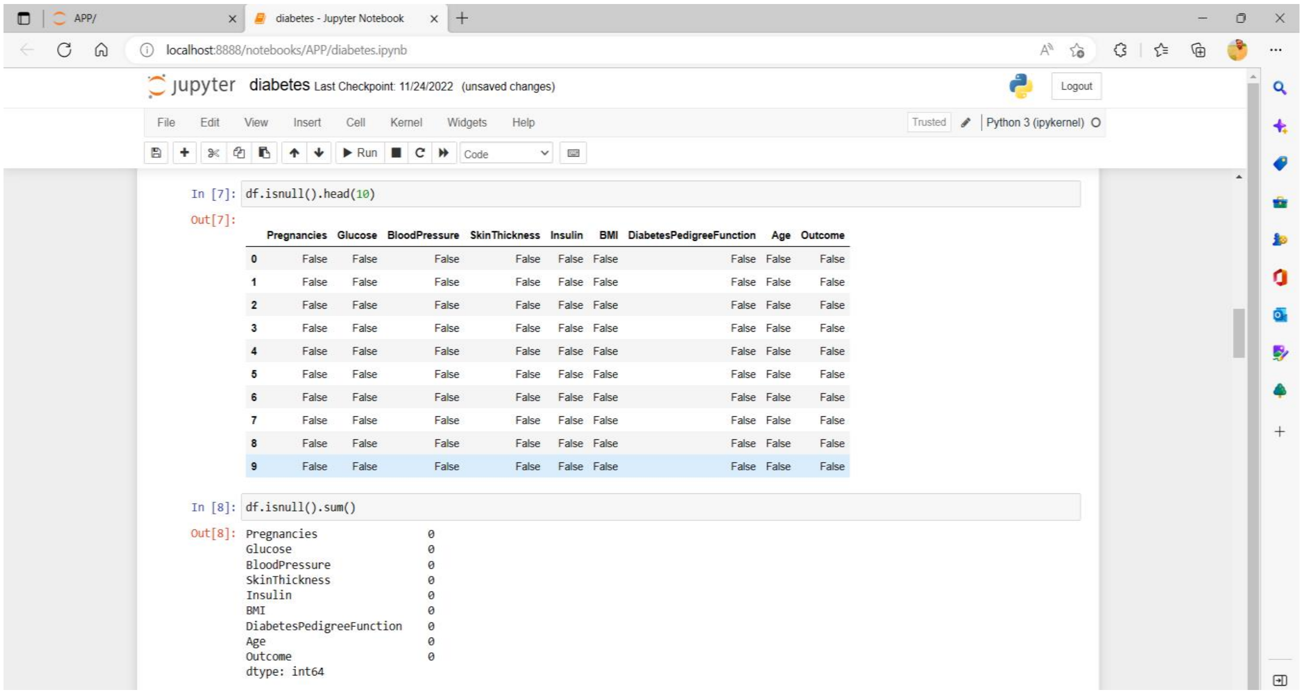
```
In [5]: df.describe()
Out[5]:
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

```
In [6]: df.describe().T
Out[6]:
```

	count	mean	std	min	25%	50%	75%	max
Pregnancies	768.0	3.845052	3.369578	0.000	1.00000	3.0000	6.00000	17.00
Glucose	768.0	120.894531	31.972618	0.000	99.00000	117.0000	140.25000	199.00
BloodPressure	768.0	69.105469	19.355807	0.000	62.00000	72.0000	80.00000	122.00
SkinThickness	768.0	20.536458	15.952218	0.000	0.00000	23.0000	32.00000	99.00
Insulin	768.0	79.799479	115.244002	0.000	0.00000	30.5000	127.25000	846.00
BMI	768.0	31.992578	7.884160	0.000	27.30000	32.0000	36.60000	67.10
DiabetesPedigreeFunction	768.0	0.471876	0.331329	0.078	0.24375	0.3725	0.62625	2.42
Age	768.0	33.240885	11.760232	21.000	24.00000	29.0000	41.00000	81.00
Outcome	768.0	0.348958	0.476951	0.000	0.00000	0.0000	1.00000	1.00

Fig 4.3: modifying of code

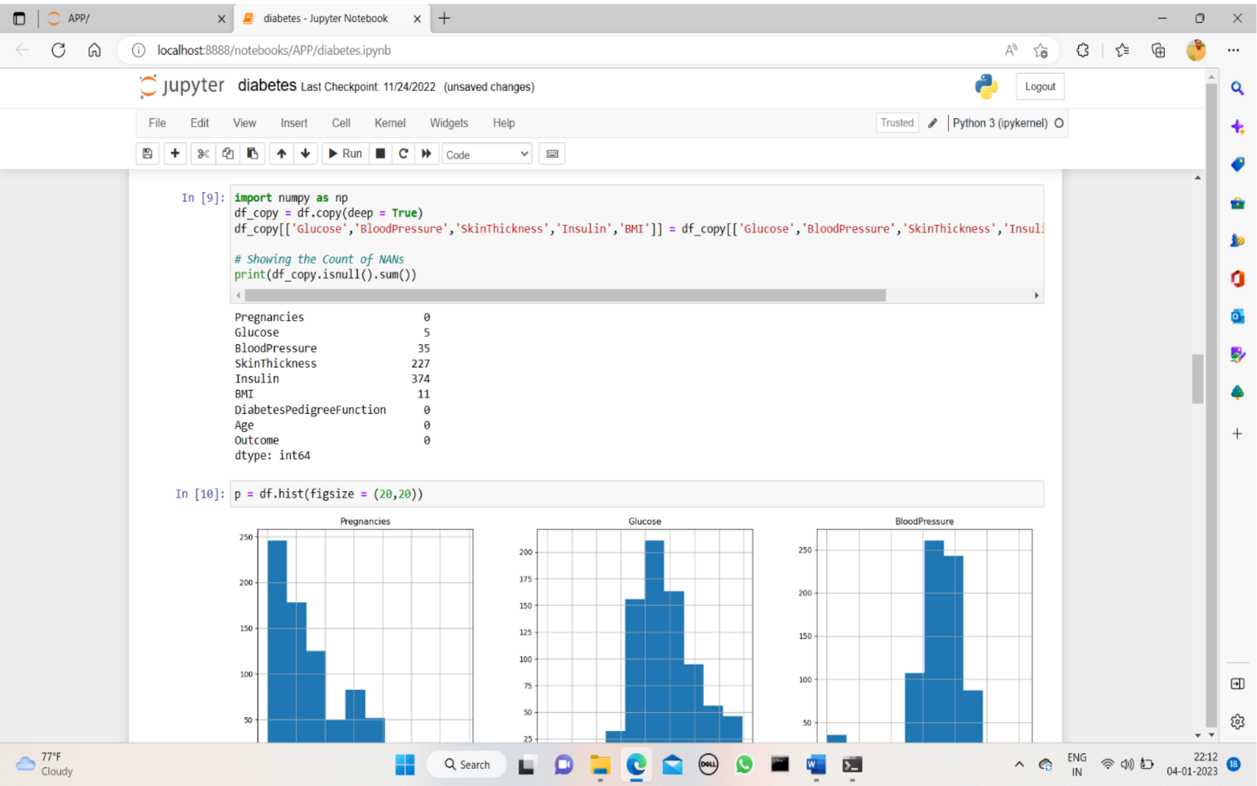

APP/
diabetes - Jupyter Notebook
localhost:8888/notebooks/APP/diabetes.ipynb
jupyter diabetes Last Checkpoint: 11/24/2022 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [7]: df.isnull().head(10)
Out[7]:
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False	False	False
7	False	False	False	False	False	False	False	False	False
8	False	False	False	False	False	False	False	False	False
9	False	False	False	False	False	False	False	False	False

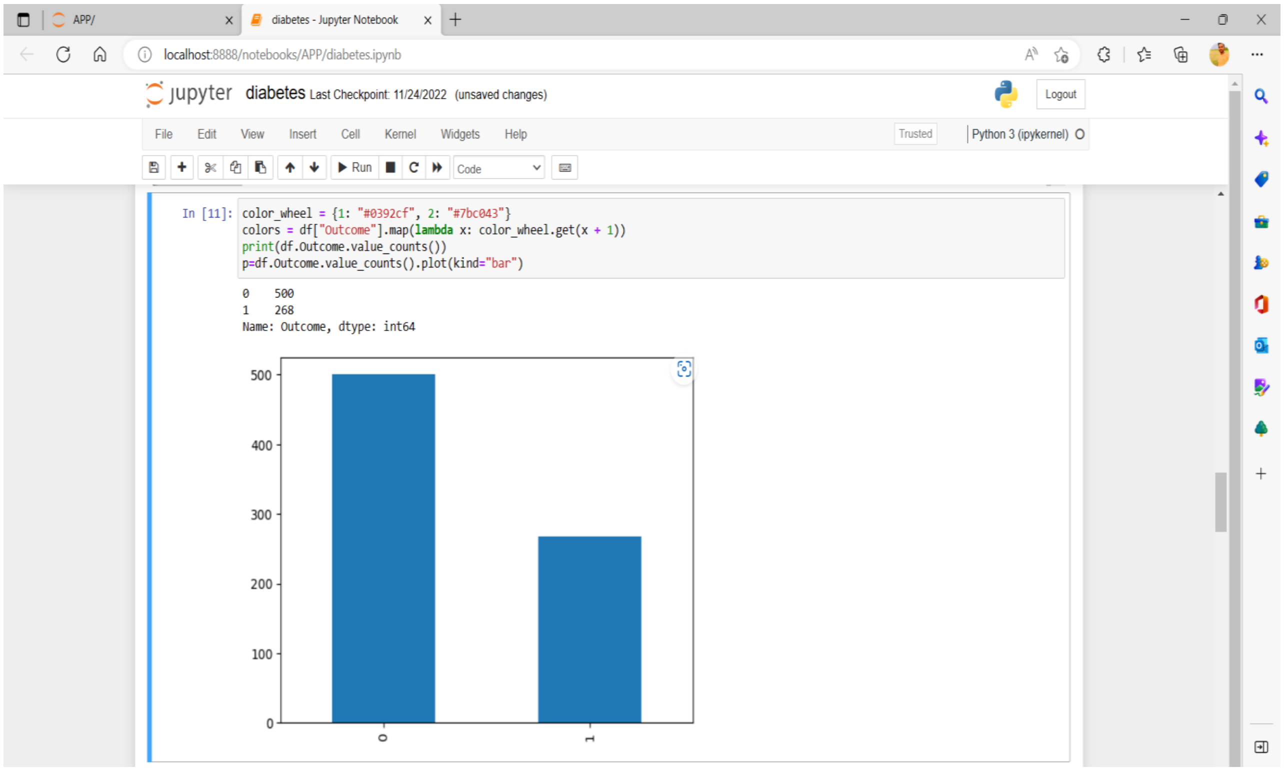
```
In [8]: df.isnull().sum()
Out[8]: Pregnancies                 0
        Glucose                     0
        BloodPressure               0
        SkinThickness               0
        Insulin                     0
        BMI                         0
        DiabetesPedigreeFunction    0
        Age                         0
        Outcome                     0
        dtype: int64
```

Fig 4.4: retrieveing of data



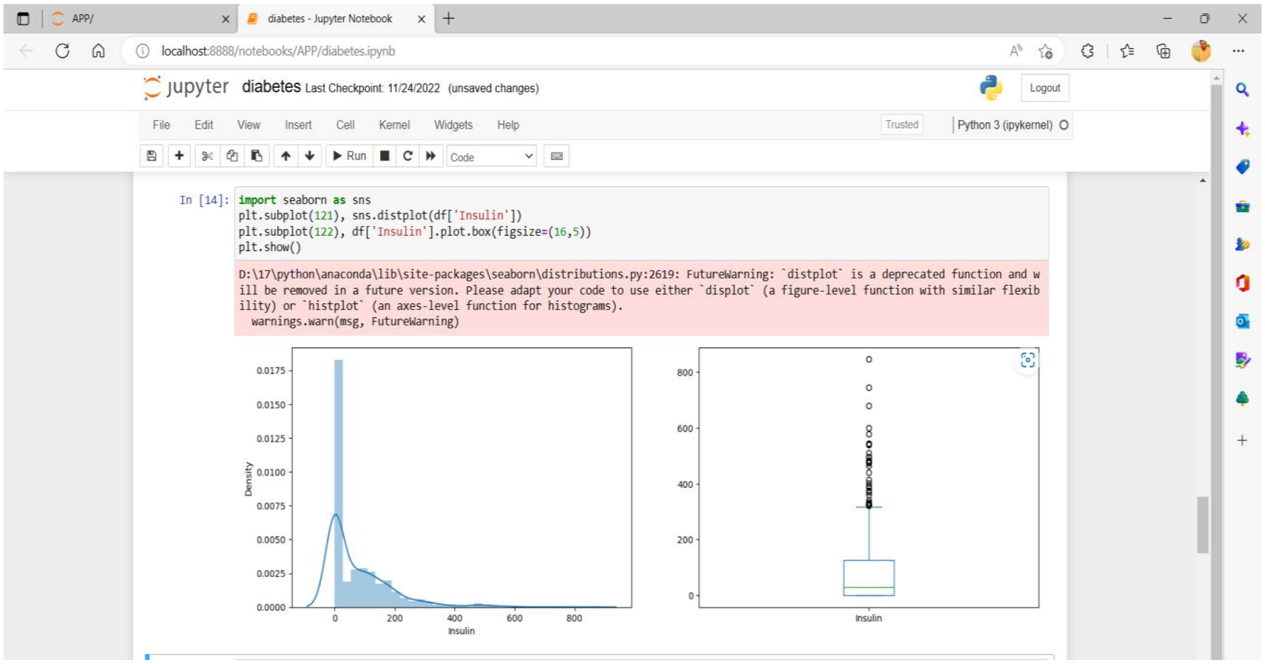
```
In [9]: import numpy as np
        df_copy = df.copy(deep = True)
        df_copy[['Glucose','BloodPressure','SkinThickness','Insulin','BMI']] = df_copy[['Glucose','BloodPressure','SkinThickness','Insuli

        # Showing the Count of NANs
        print(df_copy.isnull().sum())

        Pregnancies                   0
        Glucose                       5
        BloodPressure                35
        SkinThickness               227
        Insulin                     374
        BMI                          11
        DiabetesPedigreeFunction      0
        Age                           0
        Outcome                       0
        dtype: int64

In [10]: p = df.hist(figsize = (20,20))
```

Fig 4.5: histogram


APP/
diabetes - Jupyter Notebook
localhost:8888/notebooks/APP/diabetes.ipynb
jupyter
diabetes Last Checkpoint: 11/24/2022 (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code
500
400
300
200
100
0
0
1

```
In [11]: color_wheel = {1: "#0392cf", 2: "#7bc043"}
         colors = df["Outcome"].map(lambda x: color_wheel.get(x + 1))
         print(df.Outcome.value_counts())
         p=df.Outcome.value_counts().plot(kind="bar")

         0    500
         1    268
         Name: Outcome, dtype: int64
```

Fig 4.6: box plot

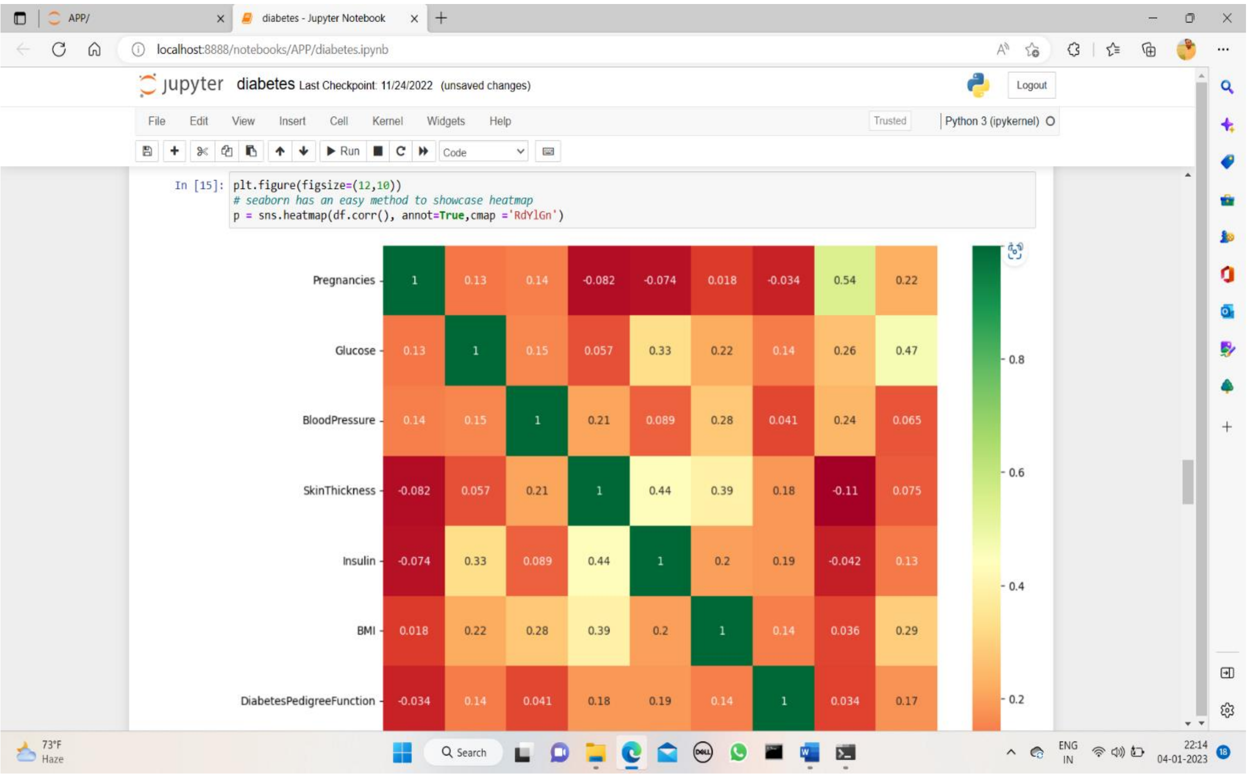


In [14]:

```
import seaborn as sns
plt.subplot(121), sns.distplot(df['Insulin'])
plt.subplot(122), df['Insulin'].plot.box(figsize=(16,5))
plt.show()
```

```
D:\17\python\anaconda\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and w
ill be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexib
ility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Fig 4.7: dist plot

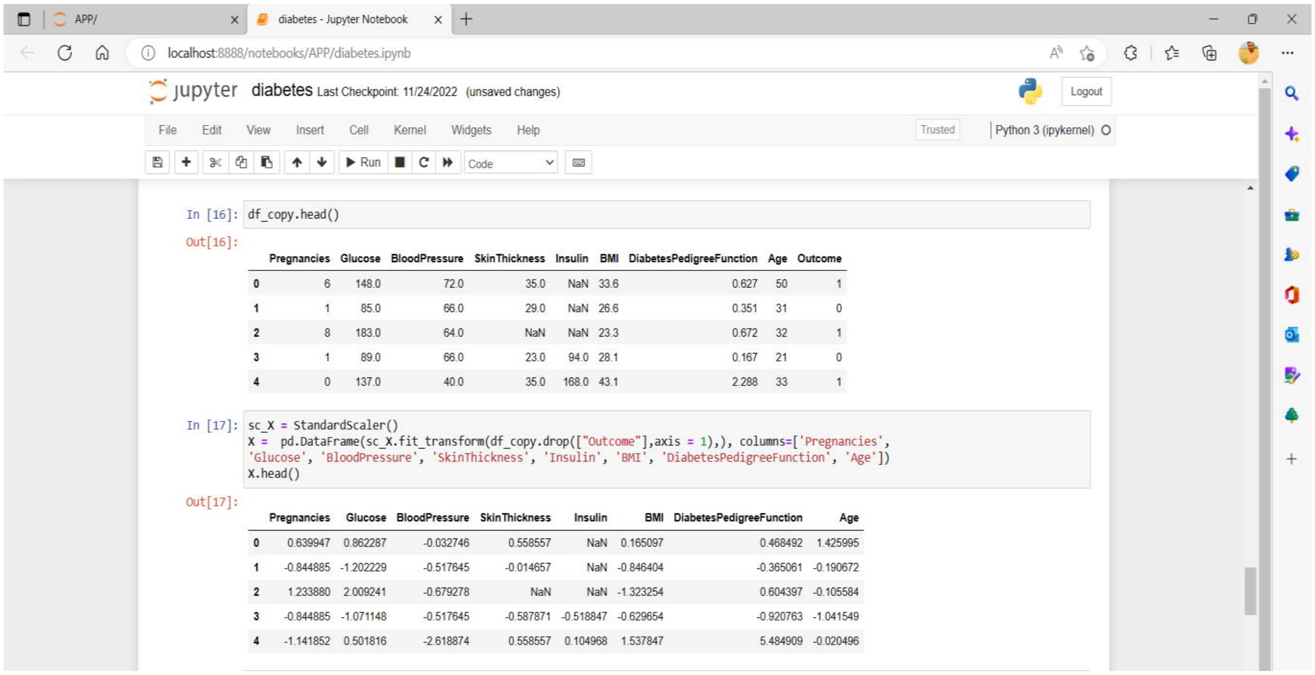


In [15]:

```
plt.figure(figsize=(12,10))
# seaborn has an easy method to showcase heatmap
p = sns.heatmap(df.corr(), annot=True,cmap ='RdYlGn')
```

Pregnancies	1	0.13	0.14	-0.082	-0.074	0.018	-0.034	0.54	0.22
Glucose	0.13	1	0.15	0.057	0.33	0.22	0.14	0.26	0.47
BloodPressure	0.14	0.15	1	0.21	0.089	0.28	0.041	0.24	0.065
SkinThickness	-0.082	0.057	0.21	1	0.44	0.39	0.18	-0.11	0.075
Insulin	-0.074	0.33	0.089	0.44	1	0.2	0.19	-0.042	0.13
BMI	0.018	0.22	0.28	0.39	0.2	1	0.14	0.036	0.29
DiabetesPedigreeFunction	-0.034	0.14	0.041	0.18	0.19	0.14	1	0.034	0.17

Fig 4.8: heatmap


APP/
diabetes - Jupyter Notebook
localhost:8888/notebooks/APP/diabetes.ipynb
jupyter
diabetes Last Checkpoint: 11/24/2022 (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code

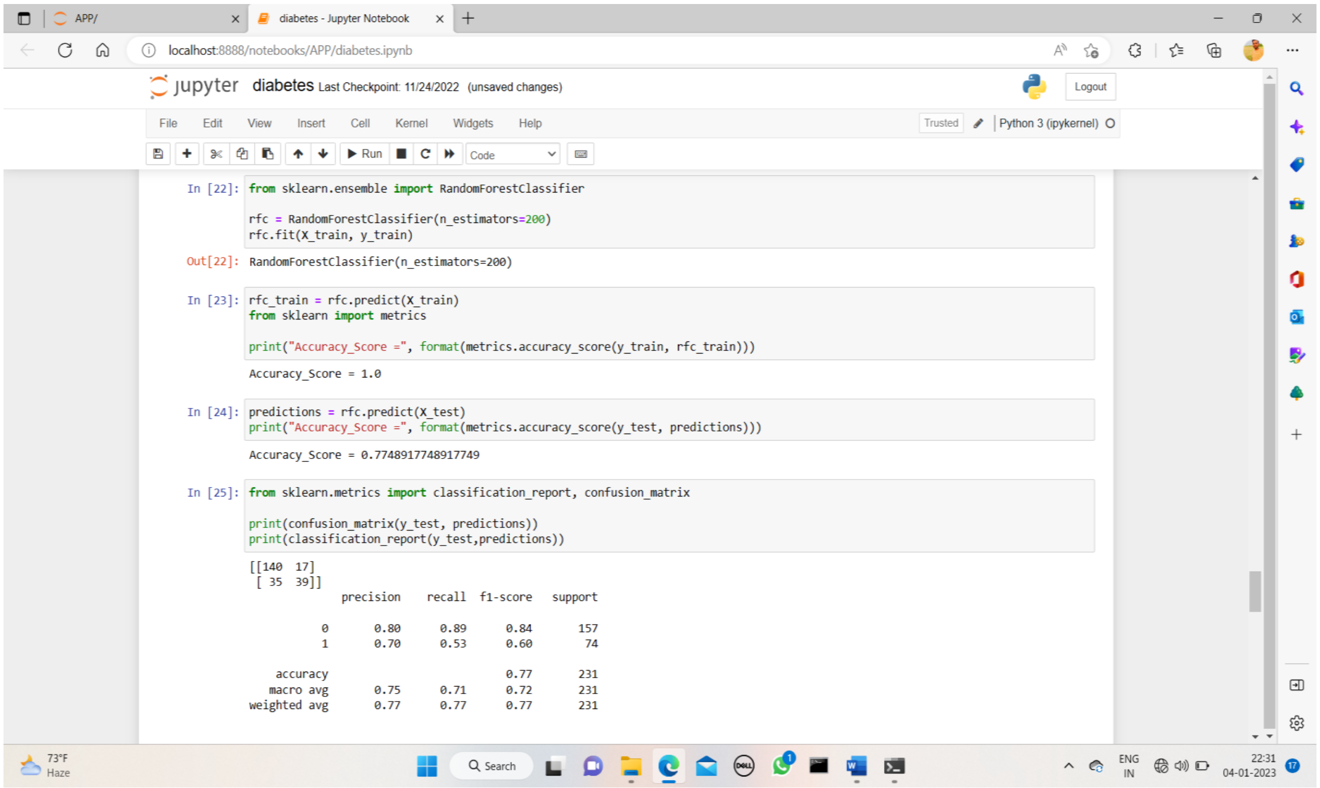
```
In [16]: df_copy.head()
Out[16]:
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148.0	72.0	35.0	NaN	33.6	0.627	50	1
1	1	85.0	66.0	29.0	NaN	26.6	0.351	31	0
2	8	183.0	64.0	NaN	NaN	23.3	0.672	32	1
3	1	89.0	66.0	23.0	94.0	28.1	0.167	21	0
4	0	137.0	40.0	35.0	168.0	43.1	2.288	33	1

```
In [17]: sc_X = StandardScaler()
         X =  pd.DataFrame(sc_X.fit_transform(df_copy.drop(["Outcome"],axis = 1),), columns=['Pregnancies',
         'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age'])
         X.head()
Out[17]:
```

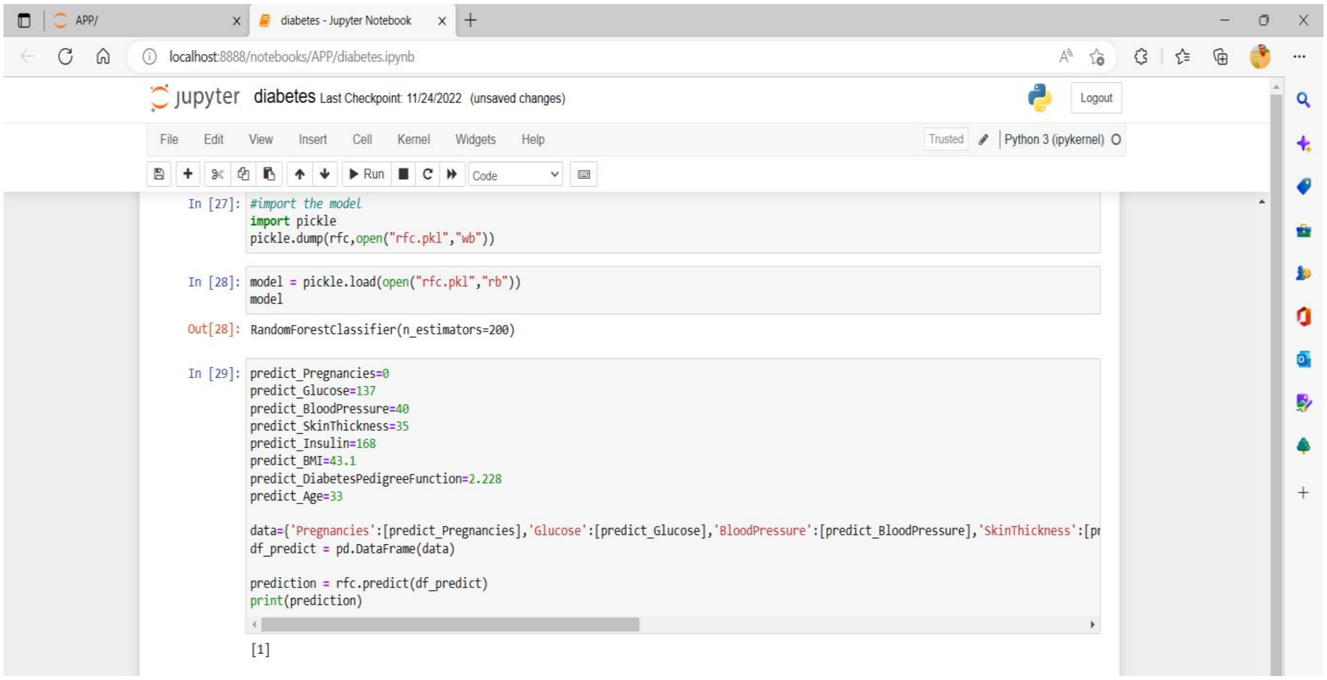
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	0.639947	0.862287	-0.032746	0.558557	NaN	0.165097	0.468492	1.425995
1	-0.844885	-1.202229	-0.517645	-0.014657	NaN	-0.846404	-0.365061	-0.190672
2	1.233880	2.009241	-0.679278	NaN	NaN	-1.323254	0.604397	-0.105584
3	-0.844885	-1.071148	-0.517645	-0.587871	-0.518847	-0.629654	-0.920763	-1.041549
4	-1.141852	0.501816	-2.618874	0.558557	0.104968	1.537847	5.484909	-0.020496

Fig 4.9: declaring of x and y variables


APP/
diabetes - Jupyter Notebook
localhost:8888/notebooks/APP/diabetes.ipynb
jupyter
diabetes Last Checkpoint: 11/24/2022 (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code
73°F
Haze
Search
ENG
IN
22:31
04-01-2023

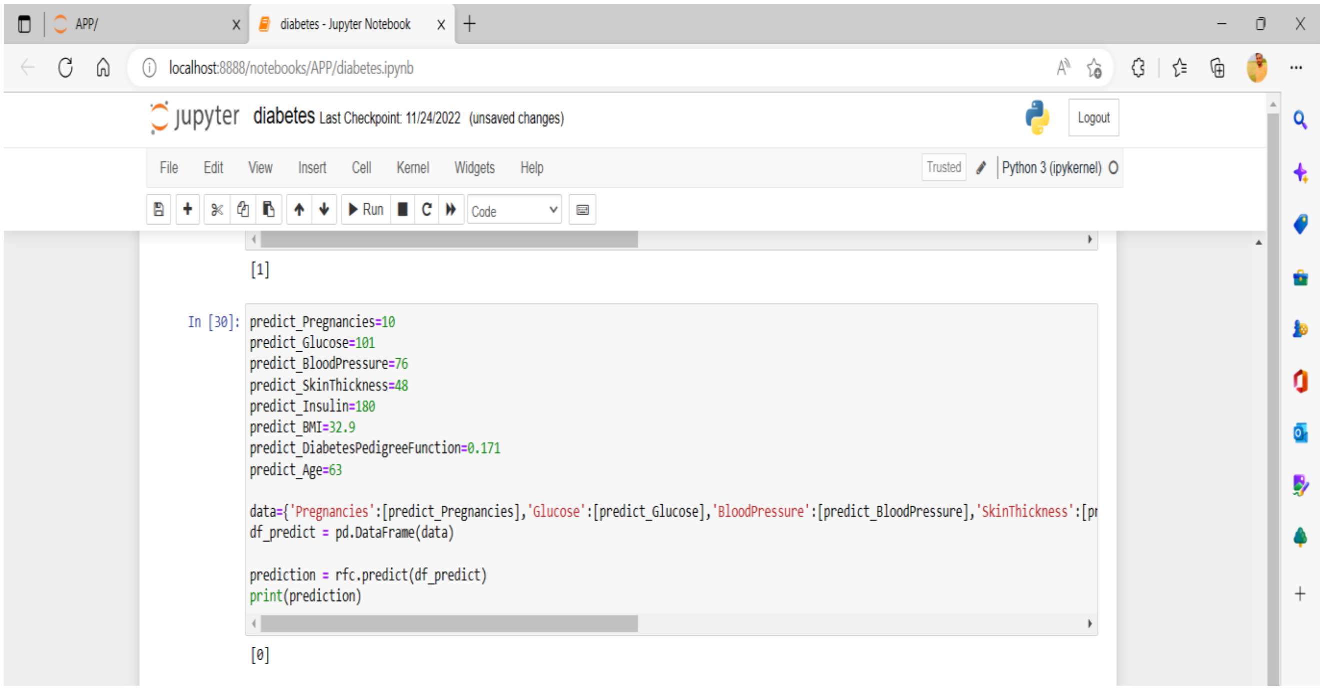
```
In [22]: from sklearn.ensemble import RandomForestClassifier

         rfc = RandomForestClassifier(n_estimators=200)
         rfc.fit(X_train, y_train)

Out[22]: RandomForestClassifier(n_estimators=200)

In [23]: rfc_train = rfc.predict(X_train)
         from sklearn import metrics

         print("Accuracy_Score =", format(metrics.accuracy_score(y_train, rfc_train)))

         Accuracy_Score = 1.0

In [24]: predictions = rfc.predict(X_test)
         print("Accuracy_Score =", format(metrics.accuracy_score(y_test, predictions)))

         Accuracy_Score = 0.7748917748917749

In [25]: from sklearn.metrics import classification_report, confusion_matrix

         print(confusion_matrix(y_test, predictions))
         print(classification_report(y_test,predictions))

         [[140  17]
          [ 35  39]]
                       precision    recall  f1-score   support

                    0       0.80      0.89      0.84       157
                    1       0.70      0.53      0.60        74

             accuracy                           0.77       231
            macro avg       0.75      0.71      0.72       231
         weighted avg       0.77      0.77      0.77       231
```

Fig 4.10: importing random forest classifier


APP/
diabetes - Jupyter Notebook
localhost:8888/notebooks/APP/diabetes.ipynb
jupyter diabetes Last Checkpoint: 11/24/2022 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [27]: #import the model
         import pickle
         pickle.dump(rfc,open("rfc.pkl","wb"))

In [28]: model = pickle.load(open("rfc.pkl","rb"))
         model

Out[28]: RandomForestClassifier(n_estimators=200)

In [29]: predict_Pregnancies=0
         predict_Glucose=137
         predict_BloodPressure=40
         predict_SkinThickness=35
         predict_Insulin=168
         predict_BMI=43.1
         predict_DiabetesPedigreeFunction=2.228
         predict_Age=33

         data={'Pregnancies':[predict_Pregnancies],'Glucose':[predict_Glucose],'BloodPressure':[predict_BloodPressure],'SkinThickness':[pr
         df_predict = pd.DataFrame(data)

         prediction = rfc.predict(df_predict)
         print(prediction)

         [1]
```

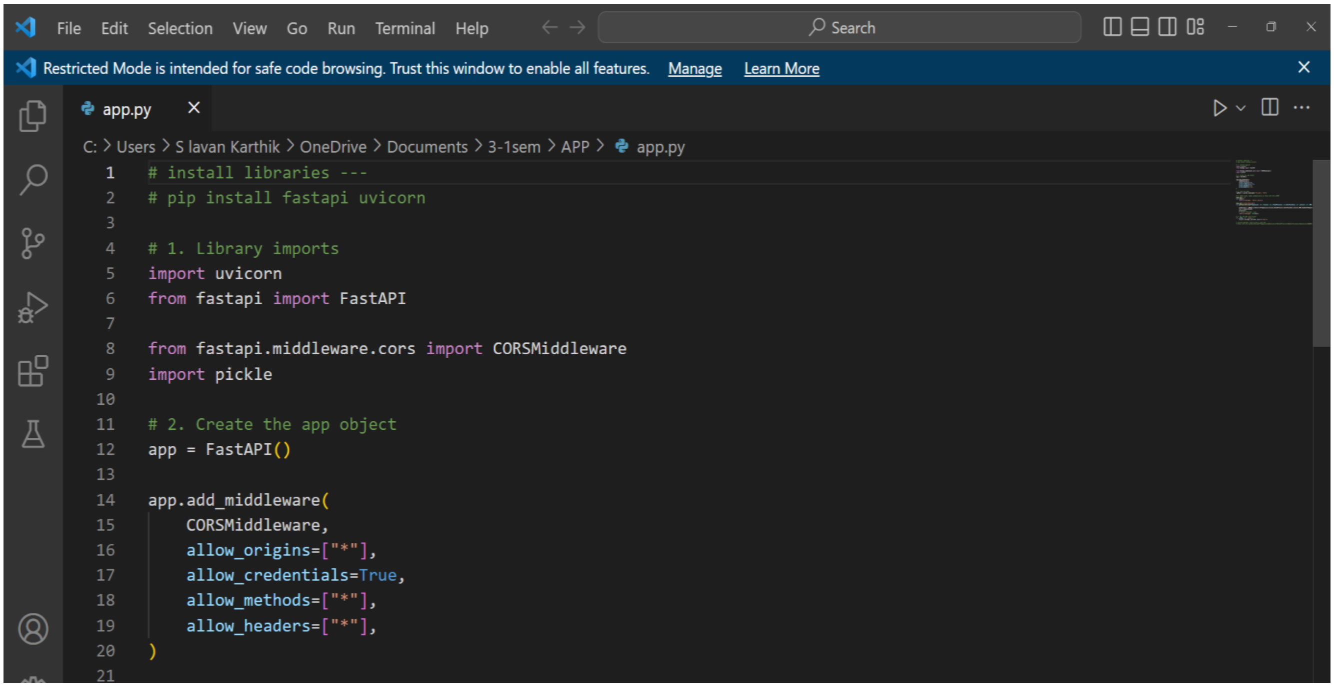
Fig 4.11: import pickle file


APP/
diabetes - Jupyter Notebook
localhost:8888/notebooks/APP/diabetes.ipynb
jupyter diabetes Last Checkpoint: 11/24/2022 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
         [1]

In [30]: predict_Pregnancies=10
         predict_Glucose=101
         predict_BloodPressure=76
         predict_SkinThickness=48
         predict_Insulin=180
         predict_BMI=32.9
         predict_DiabetesPedigreeFunction=0.171
         predict_Age=63

         data={'Pregnancies':[predict_Pregnancies],'Glucose':[predict_Glucose],'BloodPressure':[predict_BloodPressure],'SkinThickness':[pr
         df_predict = pd.DataFrame(data)

         prediction = rfc.predict(df_predict)
         print(prediction)

         [0]
```

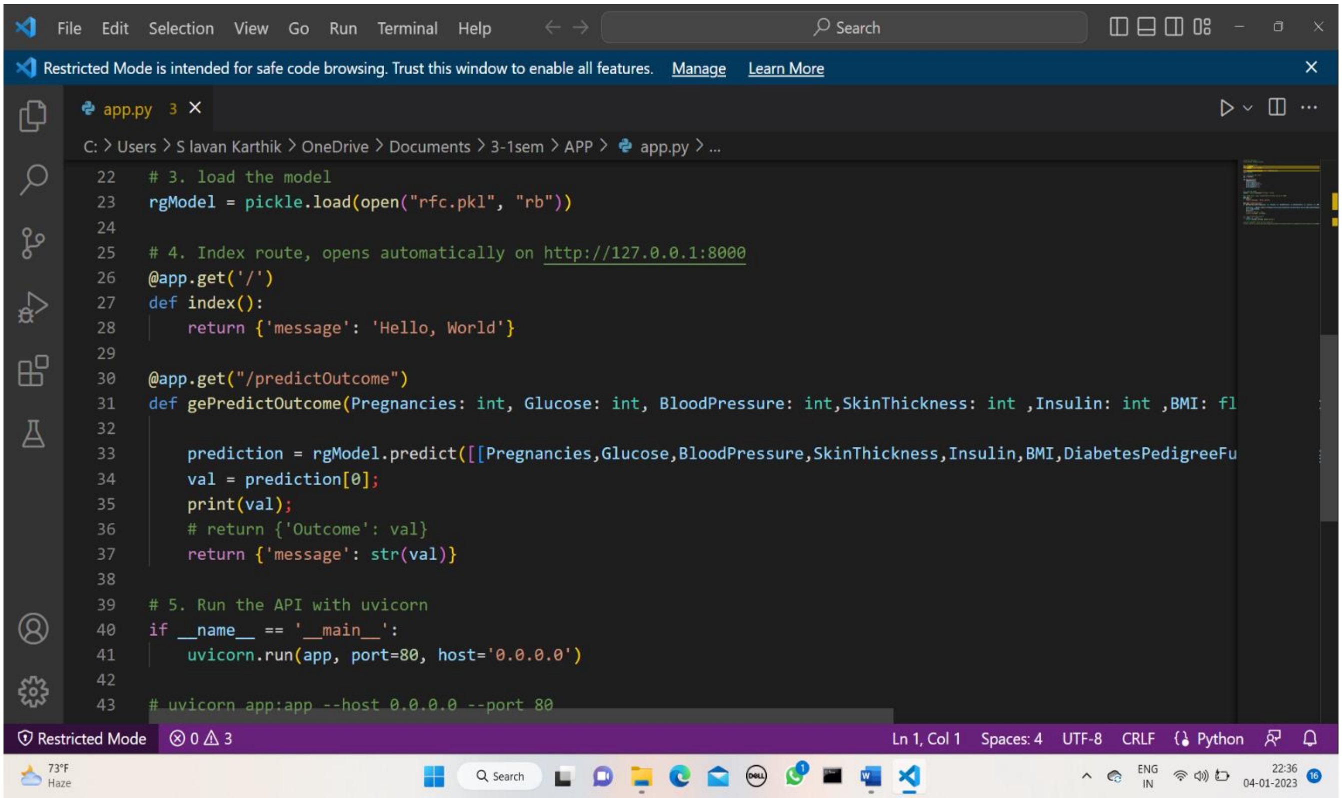
Fig 4.12: predicting diabetes

The above code is the machine learning code that has the algorithm of random forest classification with the pickle file. Then we to create the python code of FastAPI that display the machine learning code in the browser to display the output and the output is come in the webpage.


File
Edit
Selection
View
Go
Run
Terminal
Help
Search
Manage
Learn More
app.py
C: > Users > S lavan Karthik > OneDrive > Documents > 3-1sem > APP > app.py

Restricted Mode is intended for safe code browsing. Trust this window to enable all features.

```python
# install libraries ---
# pip install fastapi uvicorn

# 1. Library imports
import uvicorn
from fastapi import FastAPI

from fastapi.middleware.cors import CORSMiddleware
import pickle

# 2. Create the app object
app = FastAPI()

app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

```



Restricted Mode is intended for safe code browsing. Trust this window to enable all features.

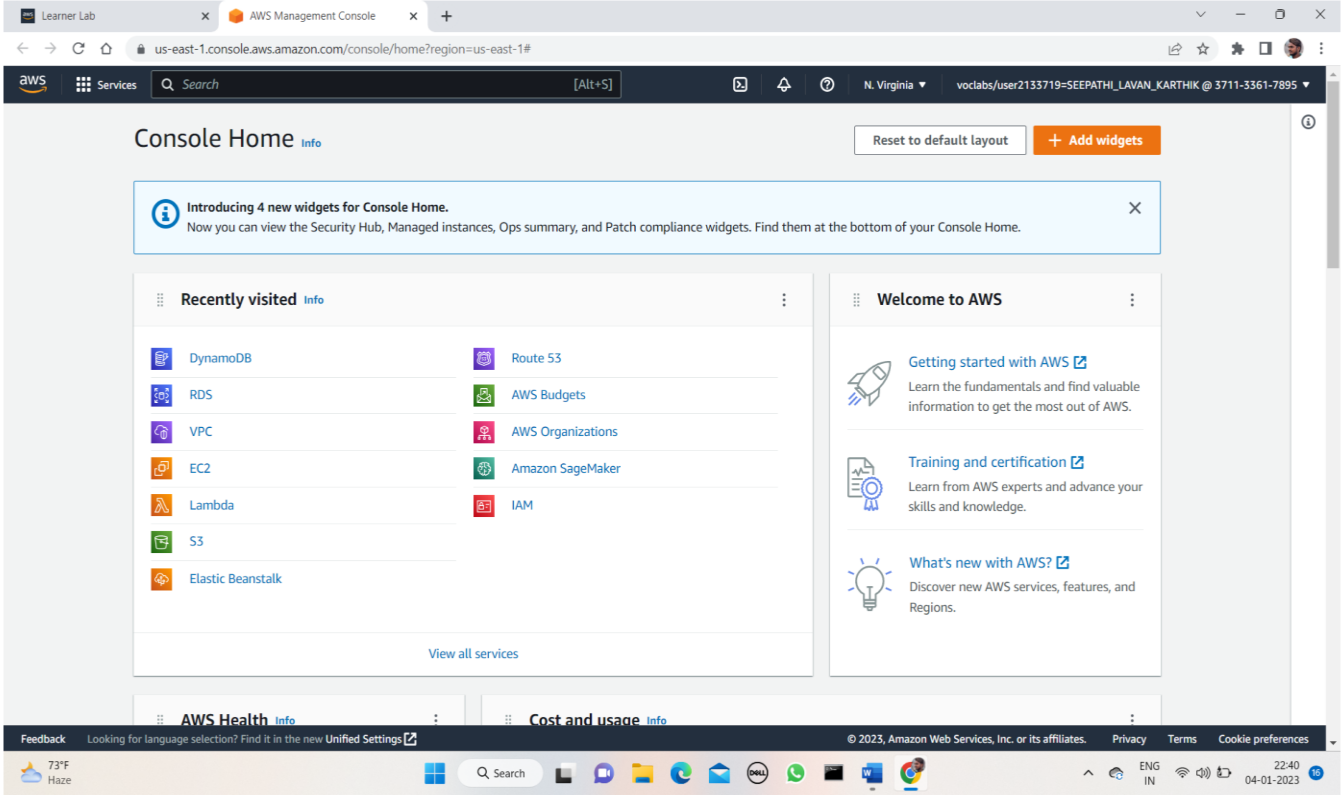
```python
# 3. load the model
rgModel = pickle.load(open("rfc.pkl", "rb"))

# 4. Index route, opens automatically on http://127.0.0.1:8000
@app.get('/')
def index():
    return {'message': 'Hello, World'}

@app.get("/predictOutcome")
def gePredictOutcome(Pregnancies: int, Glucose: int, BloodPressure: int,SkinThickness: int ,Insulin: int ,BMI: fl

    prediction = rgModel.predict([[Pregnancies,Glucose,BloodPressure,SkinThickness,Insulin,BMI,DiabetesPedigreeFu
    val = prediction[0];
    print(val);
    # return {'Outcome': val}
    return {'message': str(val)}

# 5. Run the API with uvicorn
if __name__ == '__main__':
    uvicorn.run(app, port=80, host='0.0.0.0')

# uvicorn app:app --host 0.0.0.0 --port 80
```

Fig 5: python code

After the python code run the code in the terminal with the command.

uvicorn app:app --host 0.0.0.0 –port 80 Then run the link in the browser.

Then you got the result in the browser that is output of the machine learning code. Then deploy the code in the AWS management console instance and run the IP address.


Learner Lab
AWS Management Console
us-east-1.console.aws.amazon.com/console/home?region=us-east-1#
aws
Services
Search
[Alt+S]
N. Virginia ▾
voclabs/user2133719=SEEPATHI_LAVAN_KARTHIK @ 3711-3361-7895 ▾

Console Home Info

Reset to default layout

+ Add widgets

Introducing 4 new widgets for Console Home.
Now you can view the Security Hub, Managed instances, Ops summary, and Patch compliance widgets. Find them at the bottom of your Console Home.

Recently visited Info

DynamoDB	Route 53
RDS	AWS Budgets
VPC	AWS Organizations
EC2	Amazon SageMaker
Lambda	IAM
S3	
Elastic Beanstalk	

View all services

Welcome to AWS

Getting started with AWS
Learn the fundamentals and find valuable information to get the most out of AWS.

Training and certification
Learn from AWS experts and advance your skills and knowledge.

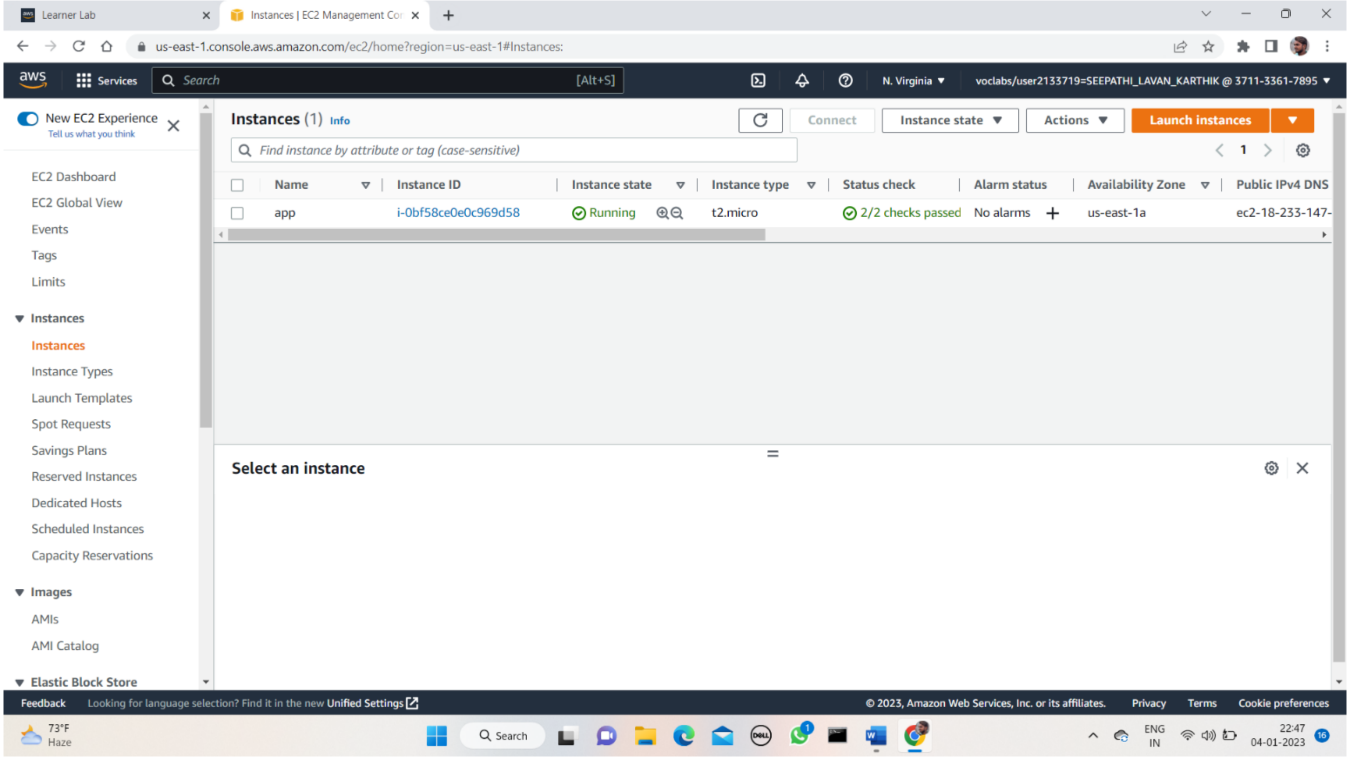
What's new with AWS?
Discover new AWS services, features, and Regions.

AWS Health Info

Cost and usage Info

Feedback
Looking for language selection? Find it in the new Unified Settings
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
73°F
Haze
Search
ENG
IN
22:40
04-01-2023

Fig 6: AWS environment



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
app	i-0bf58ce0e0c969d58	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-18-233-147-

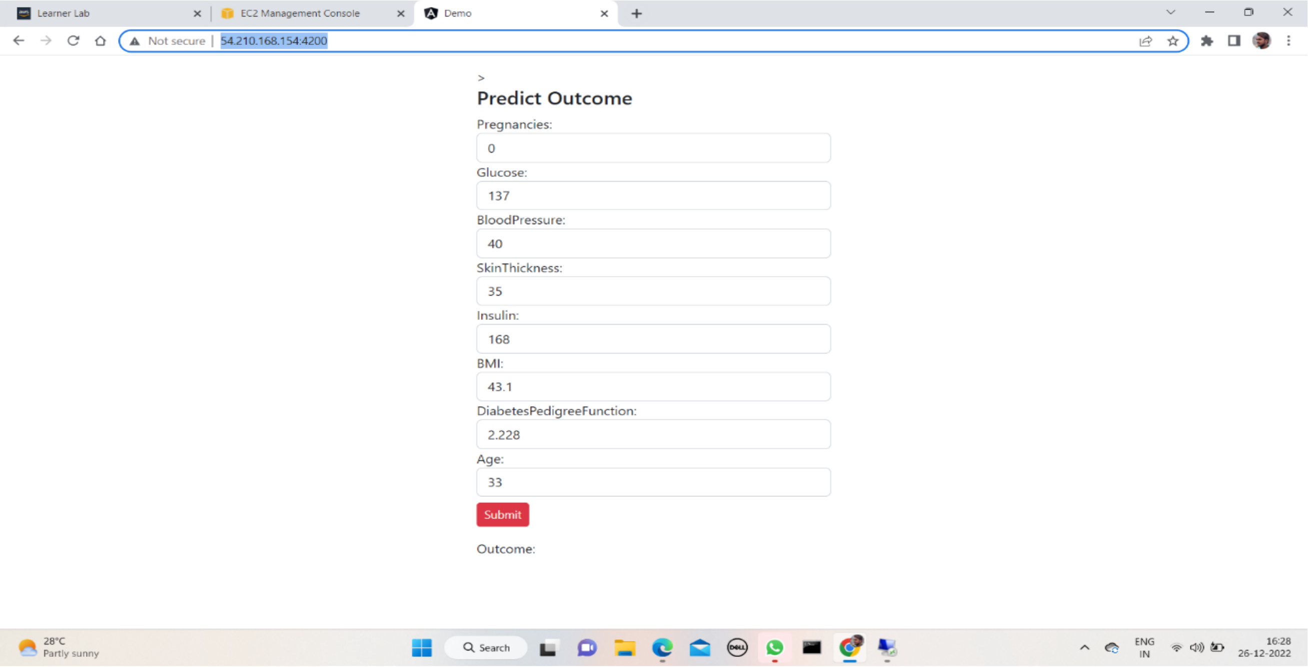
Fig 7: EC2 instance dashboard

5.OUTPUT SCREENSHOT

We use the angular for the frontend. To represent the outcome for the machine learning we use the angular application. In the angular we create the service component and we connect the machine learning model to the service by deploying the model in the AWS EC2 instance and that IP address is to be call in the service component in the angular. After completing the connection, we have to run the angular project. To check the project then after you have run the IP address then the angular project should run then after the machine learning model should run in the instances also, we can deploy the angular application in the instance as well as in the s3 bucket of the AWS management console. We see in the instances as well as in the s3 bucket also.

Deploying in the instances:

IP address should run with the port at which it has running.



>

Predict Outcome

Pregnancies: 0

Glucose: 137

BloodPressure: 40

SkinThickness: 35

Insulin: 168

BMI: 43.1

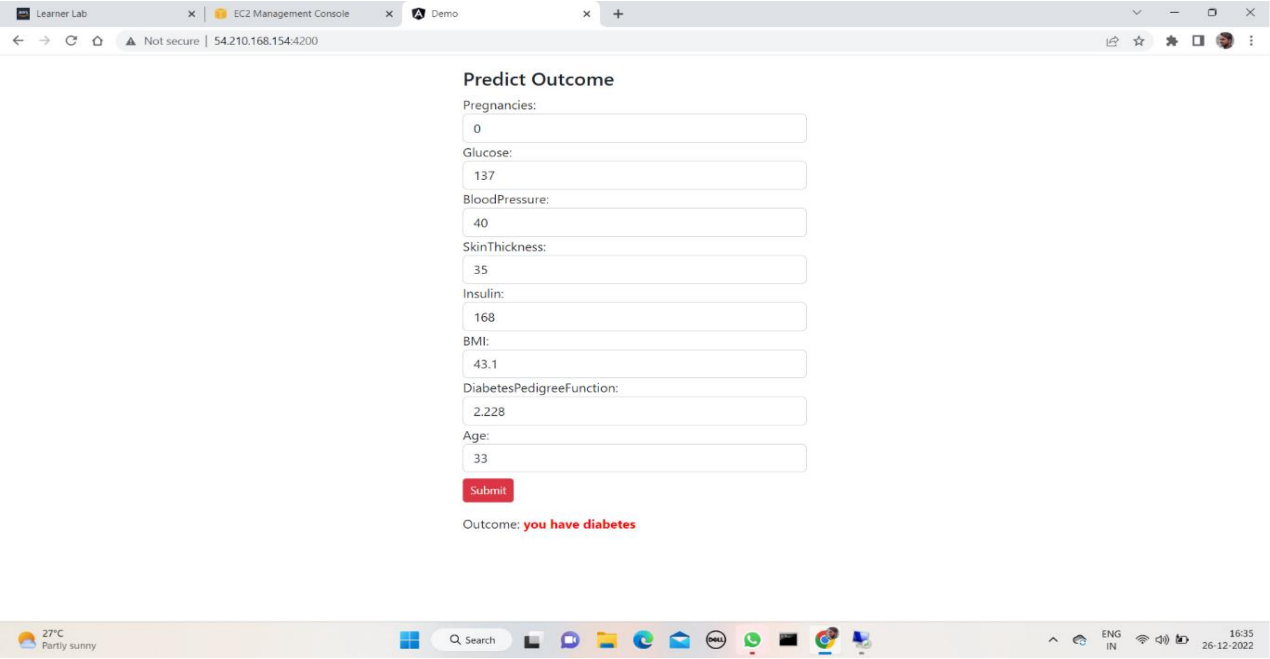
DiabetesPedigreeFunction: 2.228

Age: 33

Outcome:

Fig 8: frontend page

Case 1: If the patient has the diabetes, then the outcome would be 1. By giving input values we get the outcome.



Predict Outcome

Pregnancies: 0

Glucose: 137

BloodPressure: 40

SkinThickness: 35

Insulin: 168

BMI: 43.1

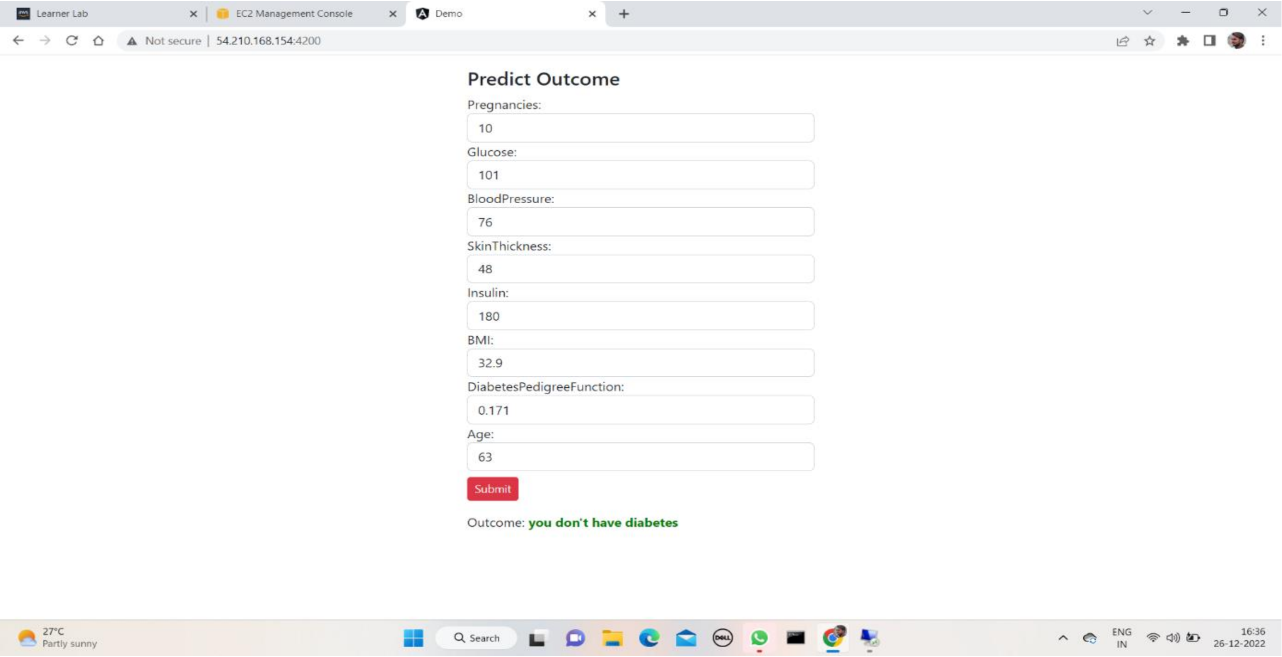
DiabetesPedigreeFunction: 2.228

Age: 33

Outcome: **you have diabetes**

Fig 9: **Outcome:** you have diabetes.

Case 2: If the patient doesn't have the diabetes, then the outcome will be. By the given input we can get the outcome.



Predict Outcome

Pregnancies: 10

Glucose: 101

BloodPressure: 76

SkinThickness: 48

Insulin: 180

BMI: 32.9

DiabetesPedigreeFunction: 0.171

Age: 63

Outcome: you don't have diabetes

Fig 10: **Outcome:** you don't have diabetes.

6.CONCLUSION AND FUTURE ENHANCEMENT

One of the important real-world medical problems is the detection of diabetes at its early stage. In this study, systematic efforts are made in designing a system which results in the prediction of diabetes. During this work, five machine learning classification algorithms are studied and evaluated on various measures. Experiments are performed on john Diabetes Database. Experimental results determine the adequacy of the designed system with an achieved accuracy of 99% using Decision Tree algorithm. In future, the designed system with the used machine learning classification algorithms can be used to predict or diagnose other diseases. The work can be extended and improved for the automation of diabetes analysis including some other machine learning algorithms.

7.REFERENCES

1. Alumax, A.A., Ahamad, M.G., Siddiqui, M.K., 2013. Application of data mining: Diabetes health care in young and old patients. Journal of King Saud University - Computer and Information Sciences 25, 127–136. doi:10.1016/j.jksuci.2012.10.003.
2. Arora, R., Suman, 2012. Comparative Analysis of Classification Algorithms on Different Datasets using WEKA. International Journal of Computer Applications 54, 21–25. doi:10.5120/8626-2492.
3. Belmonte, M.P., G.R., 2014. Design of Classifier for Detection of Diabetes Mellitus Using Genetic Programming. Advances in Intelligent Systems and Computing 1, 763–770. doi:10.1007/978-3- 319-11933-5.
4. https://www.kaggle.com/johndasilva/diabetes
5. Rani, A. S., & Jyothi, S. (2016, March). Performance analysis of classification algorithms under different datasets. In Computing for Sustainable Global Development (INDIA.Com), 2016 3rd International Conference on (pp. 1584- 1589). IEEE.