**PROJECT REPORT**

On

**SMART HEALTH PREDICTOR**

Submitted For Partial Fulfillment of Award of

**BACHELOR OF TECHNOLOGY**

**In**

**Computer Science & Engineering (2023)**

By

**Mohd Arshil Siddiqui (1901012121)**

**Tuba Parveen (1901012180)**

Under the Guidance

Of

Ms. Ankita Srivastava



**INTEGRAL UNIVERSITY, LUCKNOW (INDIA)**

**CERTIFICATE**

It is Certified that the project entitled “**SMART HEALTH PREDICTOR”** submitted by Mohd Arshil Siddiqui (1901012121) and Tuba Parveen(1901012180**)** in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (COMPUTER SCIENCE ENGINEERING) **Integral University, Lucknow (INDIA)**, is a record of students‟ own work carried under supervision and guidance of **Ms. Ankita Srivastava**. The project report embodies results of original work and studies carried out by students and the contents do not forms the basis for the award of any other degree to the candidate or to anybody else.

(Project Guide Signature) (Signature of HOD)

Ms. Ankita Srivastava Dr. Kavita Agarwal

Assistant Professor

**DECLARATION**

We hereby declare that the project entitled **“SMART HEALTH PREDICTOR”** submitted by us in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science Engineering of Integral University, Lucknow, is record of our own work carried under the supervision and guidance of Ms. Ankita Srivastava along with designation.

To the best of our knowledge this project has not been submitted to Integral University, Lucknow or any other University or Institute for the award of any degree.

**MOHD ARSHIL SIDDIQUI TUBA PARVEEN**

**1901012121 1901012180**

**ACKNOWLEDGEMENT**

I would like to thank the following people for their help in the production of this project: Ms. Ankita Srivastava, project supervisor, without whose help and support throughout, this project would not have been possible.

I choose this moment to acknowledge his contribution gratefully I would like to thank my HOD and department teachers for their support and guidance.  In this aspect, I am eternally grateful to you.

I would like to acknowledge that this project was completed entirely by me and not by someone else.

Thanking you.

**Mohd Arshil Siddiqui (1901012121)**

**Tuba Parveen (1901012180)**

**PREFACE**

This Project Report has been prepared in partial fulfilment of the requirement for the MAJOR PROJECT of the programme B.tech in Computer Science Engineering (4th year 8th sem)..For preparing the Project Report, we consulted our supervisor during the suggested time, to avail the necessary information. The blend of learning and knowledge acquired during our practical studies is presented in this Project Report.

The Project Report starts with the basic concepts of python programming , machine learning algorithms and also covers the general information of the streamlit app and its functions .In this project we have implemented many concepts of Machine Learning in Python.

Our project object is to detect whether patients have a disease or not by given a number of features from patients. The motivation of our project is to save human resources in medical centers and improve accuracy of diagnosis. In our project we used different methods to visualise data and trained machine learning model to detect disease by using Logistic Regression model and SVM.

**ABSTRACT**

Machine Learning is used across many ranges around the world. The healthcare industry is no exclusion. Machine Learning can play an essential role in predicting presence/absence of locomotors disorders, Heart diseases and more. Such information, if predicted well in advance, can provide important intuitions to doctors who can then adapt their diagnosis and dealing per patient basis. We work on predicting possible Diseases in people using Machine Learning algorithms.

In this project we perform the comparative analysis of classifier Logistic Regression. Cardiovascular disease prediction aids practitioners in making more accurate health decisions for their patients. Early detection can aid people in making lifestyle changes and, if necessary, ensuring effective medical care. Machine learning (ML) is a plausible option for reducing and understanding heart symptoms of disease. The chi-square statistical test is performed to select specific attributes from the Cleveland heart disease (HD) dataset. Support vector machine (SVM), Gaussian Naive Bayes, logistic regression, LightGBM, XGBoost, and random forest algorithm have been employed for developing heart disease risk prediction model and obtained the accuracy as 80.32%, 78.68%, 80.32%, 77.04%, 73.77%, and 88.5%, respectively. The data visualization has been generated to illustrate the relationship between the features. According to the findings of the experiments, the random forest algorithm achieves 88.5% accuracy during validation for 303 data instances with 13 selected features of the Cleveland HD dataset.

Diabetes mellitus is a group of metabolic diseases characterized by hyperglycemia resulting from defects in insulin secretion, insulin action, or both . In particular, type 2 diabetes is associated with insulin resistance (insulin action defect), i.e., where cells respond poorly to insulin, affecting their glucose intake . The diagnostic criteria established by the American Diabetes Association are: a level of glycated hemoglobin (HbA1c) greater or equal to 6.5%; basal fasting blood glucose level greater than 126 mg/dL, and; blood glucose level greater or equal to 200 mg/dL 2 h after an oral glucose tolerance test with 75 g of glucose .

Diabetes mellitus is a global public health issue. In 2019, the International Diabetes Federation estimated the number of people living with diabetes worldwide at 463 million and the expected growth at 51% by the year 2045. Moreover, it is estimated that there is one undiagnosed person for each diagnosed person with a diabetes diagnosis .

The early diagnosis and treatment of type 2 diabetes are among the most relevant actions to prevent further development and complications like diabetic retinopathy . According to the ADDITION-Europe Simulation Model Study, an early diagnosis reduces the absolute and relative risk of suffering cardiovascular events and mortality . A sensitivity analysis on USA data proved a 25% relative reduction in diabetes-related complication rates for a 2-year earlier diagnosis.

Keywords: Logistic Regression, Python programming; confusion matrix; correlation matrix ; SVM .

**TABLE OF CONTENTS**

Title/cover page i

Certificate ii

Declaration iii

Acknowledgement iv

Preface v

Abstract vi

Table of Contents viii-ix

**1.INTRODUCTION 1-4**

1.1 Motivation for work 1

1.2 System Architecture 2

1.3 Machine Learning 3

**2. FEASIBILITY STUDY 5-6**

**3. PROPOSED METHODOLOGY 7-42**

3.1 Existing System 7

3.2 Proposed System 8

3.3 Hardware requirement 9

3.4 Software requirement 9

3.5 s/w tools required 7

3.6 Project development 11

3.7 Software Development Model 15

3.8 Design(UI/UX and Backend) 19

3.9 Coding 21

3.10 Testing 34

3.11 Implementation 36

3.12 Alogorithm 38

3.13 Maintenance 40

**4. RESULT ANALYSIS AND FUTURE WORK 43-46**

4.1 Screenshot 43

4.2 Result 45

**5. CONCLUSION 47**

List of Figures 48

Refrences and Bibliography 49

**CHAPTER-1**

**INTRODUCTION**

According to the World Health Organization, every year 12 million deaths occur worldwide due to Heart Disease. Heart disease is one of the biggest causes of morbidity and mortality among the population of the world. Prediction of cardiovascular disease is regarded as one of the most important subjects in the section of data analysis.The load of cardiovascular disease is rapidly increasing all over the world from the past few years. Many researches have been conducted in attempt to pinpoint the most influential factors of heart disease as well as accurately predict the overall risk. Heart Disease is even highlighted as a silent killer which leads to the death of the person without obvious symptoms.The early diagnosis of heart disease plays a vital role in making decisions on lifestyle changes in high-risk patients and in turn reduces the complications.

Diabetes mellitus is a group of metabolic diseases characterized by hyperglycemia resulting from defects in insulin secretion, insulin action, or both . In particular, type 2 diabetes is associated with insulin resistance (insulin action defect), i.e., where cells respond poorly to insulin, affecting their glucose intake . The diagnostic criteria established by the American Diabetes Association are: (1) a level of glycated hemoglobin (HbA1c) greater or equal to 6.5%; (2) basal fasting blood glucose level greater than 126 mg/dL, and; (3) blood glucose level greater or equal to 200 mg/dL 2 h after an oral glucose tolerance test with 75 g of glucose .

Machine learning proves to be effective in assisting in making decisions and predictions from the large quantity of data produced by the health care industry. This project aims to predict future Disease by analyzing data of patients which classifies whether they have heart disease or not using machine-learning algorithm. Machine Learning techniques can be a boon in this regard. Even though heart disease can occur in different forms, there is a common set of core risk factors that influence whether someone will ultimately be at risk for heart disease or not. By collecting the data from various sources, classifying them under suitable headings & finally analysing to extract the desired data we can say that this technique can be very well adapted to do the prediction of heart disease.

**1.1 MOTIVATION FOR WORK**

The main motivation of doing this research is to present a disease prediction model for the prediction of occurrence of disease. Further, this research work is aimed towards identifying the best classification algorithm for identifying the possibility of disease in a patient. This work is justified by performing a comparative study and analysis using algorithm namely Logistic Regression and SVM is used at different levels of evaluations. Although commonly used machine learning algorithms, the disease prediction is a vital task involving highest possible accuracy. Hence, this algorithm is evaluated at numerous levels and types of evaluation strategies. This will provide researchers and medical practitioners to establish a better.

**1.2 SYSTEM ARCHITECTURE**

The working of this system is described as follows: Dataset collection is collecting data which contains patient details. Attributes selection process selects the useful attributes for the prediction of disease. After identifying the available data resources, they are further selected, cleaned, made into the desired form. Different classification techniques as stated will be applied on preprocessed data to predict the accuracy of disease. Accuracy measure compares the accuracy of different classifiers.

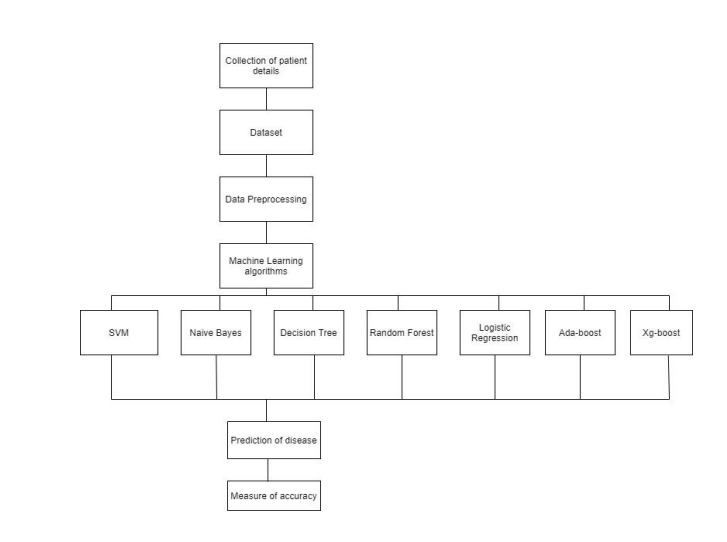
****

Figure 1.1

**1.3 MACHINE LEARNING**

In machine learning, classification refers to a predictive modeling problem where a class label is predicted for a given example of input data.

● Supervised Learning Supervised learning is the type of machine learning in which machines are trained using well "labelled" training data, and on the basis of that data, machines predict the output. The labelled data means some input data is already tagged with the correct output. In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the supervision of the teacher. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y).

● Unsupervised learning Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.

• Unsupervised learning is helpful for finding useful insights from the data.

• Unsupervised learning is much similar to how a human learns to think by their own experiences, which makes it closer to the real AI.

• Unsupervised learning works on unlabeled and uncategorized data which make unsupervised learning more important.

• In real-world, we do not always have input data with the corresponding output so to solve such cases, we need unsupervised learning.

● Reinforcement learning

Reinforcement learning is an area of Machine Learning. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behaviour or path it should take in a specific situation. Reinforcement learning differs from supervised learning in a way that in supervised learning the training data has the answer key with it so the model is 11 trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience.

**CHAPTER-2**

**FEASIBILITY STUDY**

The major challenge in disease is its detection. There are instruments available which can predict any disease but either it are expensive or are not efficient to calculate chance of heart disease in human. Early detection of cardiac diseases can decrease the mortality rate and overall complications. However, it is not possible to monitor patients everyday in all cases accurately and consultation of a patient for 24 hours by a doctor is not available since it requires more sapience, time and expertise. Since we have a good amount of data in today’s world, we can use various machine learning algorithms to analyze the data for hidden patterns. The hidden patterns can be used for health diagnosis in medicinal data.

In understanding diabetes and how it develops, we need to understand what happens in the body without diabetes. Sugar (glucose) comes from the foods that we eat, specifically carbohydrate foods. Carbohydrate foods provide our body with its main energy source everybody, even those people with diabetes, needs carbohydrate. Carbohydrate foods include bread, cereal, pasta, rice, fruit, dairy products and vegetables (especially starchy vegetables). When we eat these foods, the body breaks them down into lucose. The glucose moves around the body in the

bloodstream. Some of the glucose is taken to our brain to help us think clearly and function.

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, and to uncover key insights in data mining projects.

Machine learning algorithms are typically created using frameworks that accelerate solution development, such as TensorFlow and PyTorch.

The main purpose of the proposed method is to predict the occurrence of disease for early detection of the disease in a short time. In our approach, we are using different data mining techniques and machine learning algorithms, Naïve Bayes, k Nearest Neighbor (KNN), Decision tree, Artificial Neural Network (ANN), Random Forest to predict the heart disease based on some health parameters.

**CHAPTER-3**

**PROPOSED METHODOLOGY/ PROJECT REQUIREMENT**

**3.1 EXISTING SYSTEM**

Heart disease is even being highlighted as a silent killer which leads to the death of a person without obvious symptoms. The nature of the disease is the cause of growing anxiety about the disease & its consequences. Hence continued efforts are being done to predict the possibility of this deadly disease in prior. So that various tools & techniques are regularly being experimented with to suit the present-day health needs. Machine Learning techniques can be a boon in this regard. Even though heart disease can occur in different forms, there is a common set of core risk factors that influence whether someone will ultimately be at risk for heart disease or not. By collecting the data from various sources, classifying them under suitable headings & finally analysing to extract the desired data we can conclude. This technique can be very well adapted to the do the prediction of heart disease. As the well-known quote says “Prevention is better than cure”, early prediction & its control can be helpful to prevent & decrease the death rates due to heart disease. the classification on diverse types of datasets that can be accomplished to decide if a person is diabetic or not.

The diabetic patient’s data set is established by gathering data from hospital

warehouse which contains two hundred instances with nine attributes. These instances of this dataset are referring to two groups i.e. blood tests and urine

tests. In this study the implementation can be done by using WEKA to classify the data and the data is assessed by means of 10-fold cross validation approach, as it performs very well on small datasets, and the outcomes are compared. The naïve Bayes, J48, REP Tree and Random Tree are used. It was

concluded that J48 works best showing an accuracy of 60.2% among others.

**3.2 PROPOSED SYSTEM**

The working of the system starts with the collection of data and selecting the important attributes. Then the required data is preprocessed into the required format. The data is then divided into two parts training and testing data. The algorithms are applied and the model is trained using the training data. The accuracy of the system is obtained by testing the system using the testing data. This system is implemented using the following modules.

1.) Collection of Dataset

2.) Selection of attributes

3.) Data Pre-Processing

4.) Balancing of Data

5.) Disease Prediction

Collection of dataset

Initially, we collect a dataset for our heart disease and diabetes prediction system. After the collection of the dataset, we split the dataset into training data and testing data. The training dataset is used for prediction model learning and testing data is used for evaluating the prediction model. For this project, 80% of training data is used and 20% of data is used for testing. The dataset used for this project is Heart Disease UCI and . The dataset consists of 76 attributes; out of which, 14 attributes are used for the system.

The diabetes data set was originated from: <https://www.kaggle.com/johndasilva/diabetes>

The heart disease data set was originated from:

<https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset>

Diabetes dataset containing 2000 cases. The objective is to predict based on the measures to predict if the patient is diabetic or not.

**3.3 HARDWARE REQUIREMENTS**

Processer : Any Update Processer

Ram : Min 4GB

Hard Disk : Min 100GB

**3.4 SOFTWARE REQUIREMENTS**

Operating System : Windows family

Technology :

* Python3.9
* Streamlit

IDE : Visual Code Studio

**3.5 S/W TOOLS REQUIRED**

**Python**

Python is an interpreted, high-level, general purpose programming language created by Guido Van Rossum and first released in 1991, Python's design philosophy emphasizes code Readability with its notable use of significant White space. Its language constructs and object oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

**Sklearn**

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

**Numpy**

NumPy is a library for the python programming language, adding support for large, multi- dimensional arrays and matrices, along with a large collection of high level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim with contributions from several other developers. In 2005, Travis created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open source software and has many contributors.

**Panda**

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.

**Matplotlib**

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a statemachine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.

**Seaborn**

Seaborn is a Python data visualization library based on matplotlib. It provides a highlevel interface for drawing attractive and informative statistical graphics. Seaborn is a library in Python predominantly used for making statistical graphics. Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data.

**SciPy**

SciPy contains modules for optimization, linearalgebra, integration, interpolation, special functions, FFT, signal and imageprocessing, ODE solvers and other tasks common in science and engineering. SciPy is also a family of conferences for users and developers of these tools: SciPy (in the United States), EuroSciPy (in Europe) and SciPy.in (in India). Enthought originated the SciPy conference in the United States and continues to sponsor many of the international conferences as well as host the SciPy website. SciPy is a scientific computation library that uses NumPy underneath. It provides more utility functions for optimization, stats and signal processing.

**Streamlit**

Streamlit is an open source app framework in Python language. It helps us create web apps for data science and machine learning in a short time. The platform uses python scripting, APIs, widgets, instant deployment, team collaboration tools, and application management solutions to help data scientists and machine learning engineers create python-based applications. Applications created using Streamlit range from applications capable of real time object detection, geographic data browsers, deep dream network debuggers, to face-GAN explorers. Frameworks compatible with Streamlit .

**3.6 PROJECT DEVELOPMENT**

**Selection of attributes**

Attribute or Feature selection includes the selection of appropriate attributes for the prediction system. This is used to increase the efficiency of the system. Various attributes of the patient like gender, chest pain type, fasting blood pressure, serum cholesterol, exang, etc are selected for the prediction. The Correlation matrix is used for attribute selection for this model.

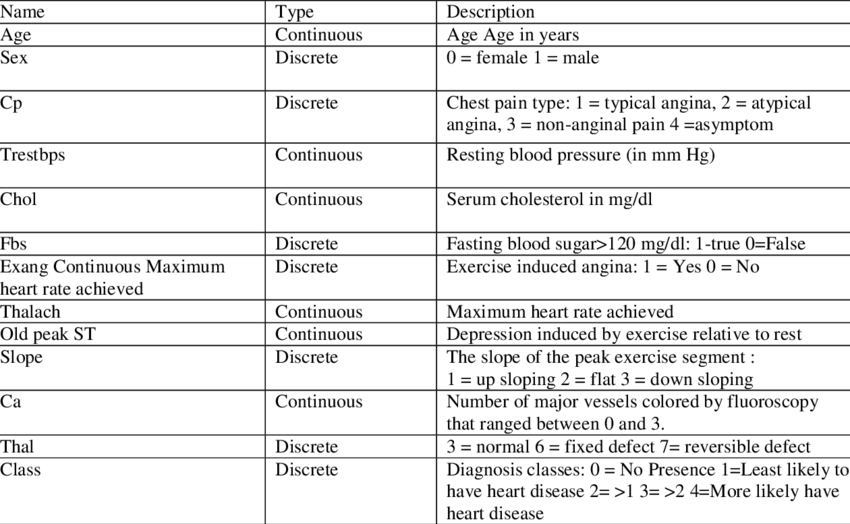


Figure 1.2

**Collection of** dataset

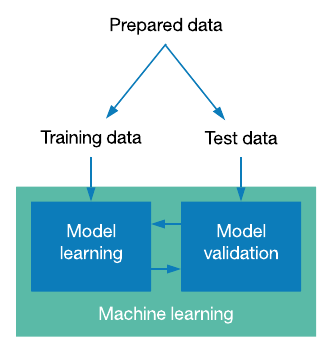
Initially, we collect a dataset for our disease prediction system. After the collection of the dataset, we split the dataset into training data and testing data. The training dataset is used for prediction model learning and testing data is used for evaluating the prediction model. For this project, 80% of training data is used and 20% of data is used for testing. The dataset used for this project is Heart Disease UCI and Diabetes dataset. The dataset consists of 76 attributes; out of which, 14 attributes are used for the system. A machine learning dataset is a collection of data that is used to train the model. A dataset acts as an example to teach the machine learning algorithm how to make predictions.

Training dataset:

This is one of the most important subsets of the whole dataset, comprising about 60% of the total dataset. This set comprises the data that will initially be used to train the model. In other words, it helps teach the algorithm what to look for in the data.

Test dataset:

This subset is input at the final stage of the training process and accounts for the last 20% of the dataset. The data in this subset is unknown to the model and is used to test the accuracy of the model.

Figure 1.3

**Pre-processing of Data**

Data pre-processing is an important step for the creation of a machine learning model. Initially, data may not be clean or in the required format for the model which can cause misleading outcomes. In pre-processing of data, we transform data into our required format. It is used to deal with noises, duplicates, and missing values of the dataset. Data pre-processing has the activities like importing datasets, splitting datasets, attribute scaling, etc. Preprocessing of data is required for improving the accuracy of the model.

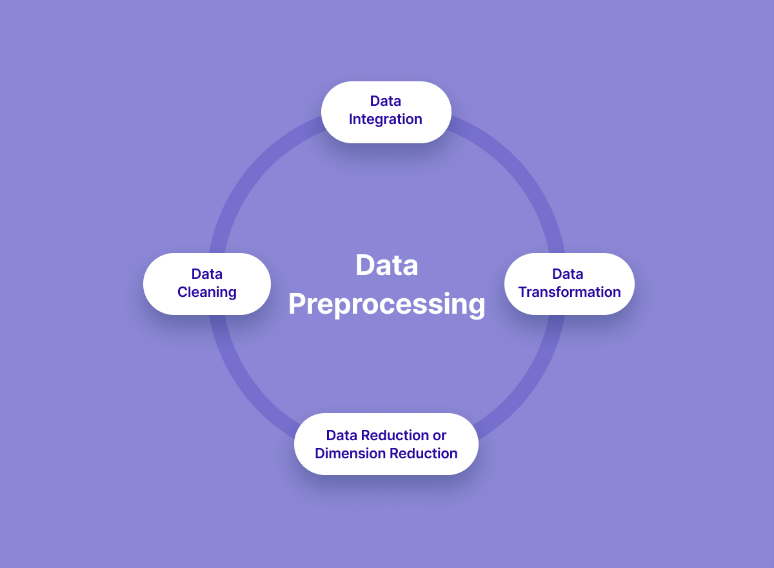


Figure 1.4

**Balancing of Data**

Imbalanced datasets can be balanced in two ways. They are Under Sampling and Over Sampling (a) Under Sampling: In Under Sampling, dataset balance is done by the reduction of the size of the ample class. This process is considered when the amount of data is adequate. (b) Over Sampling: In Over Sampling, dataset balance is done by increasing the size of the scarce samples. This process is considered when the amount of data is inadequate.

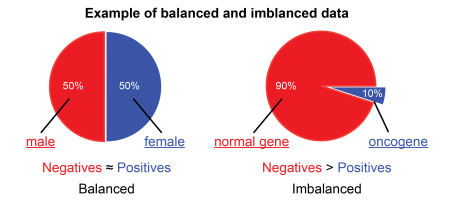


Figure 1.5

**Prediction of Disease**

Various machine learning algorithms like SVM, Naive Bayes, Decision Tree, Random Tree, Logistic Regression, Ada-boost, Xg-boost are used for classification. Comparative analysis is performed among algorithms and the algorithm that gives the highest accuracy is used for disease prediction.

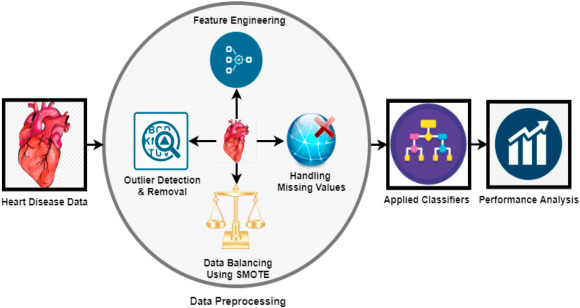
****

Figure 1.6

**3.7 SOFWATE DEVOLPMENT MODEL**

Machine learning has given the computer systems the abilities to automatically learn without being explicitly programmed. But how does a machine learning system work? So, it can be described using the life cycle of machine learning. Machine learning life cycle is a cyclic process to build an efficient machine learning project. The main purpose of the life cycle is to find a solution to the problem or project.

Machine learning life cycle involves seven major steps, which are given below:

* Gathering Data
* Data preparation
* Data Wrangling
* Analyse Data
* Train the model
* Test the model
* Deployment

Figure 1.7

The most important thing in the complete process is to understand the problem and to know the purpose of the problem. Therefore, before starting the life cycle, we need to understand the problem because the good result depends on the better understanding of the problem.

In the complete life cycle process, to solve a problem, we create a machine learning system called "model", and this model is created by providing "training". But to train a model, we need data, hence, life cycle starts by collecting data.

## Gathering Data

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.

In this step, we need to identify the different data sources, as data can be collected from various sources such as **files**, **database**, **internet**, or **mobile devices**. It is one of the most important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

This step includes the below tasks:

* **Identify various data sources**
* **Collect data**
* **Integrate the data obtained from different sources**

By performing the above task, we get a coherent set of data, also called as a **dataset**. It will be used in further steps.

## Data preparation

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

In this step, first, we put all data together, and then randomize the ordering of data.

This step can be further divided into two processes:

* **Data exploration:**  
  It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data.  
  A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.
* **Data pre-processing:**  
  Now the next step is preprocessing of data for its analysis.

## Data Wrangling

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues.

It is not necessary that data we have collected is always of our use as some of the data may not be useful. In real-world applications, collected data may have various issues, including:

* Missing Values
* Duplicate data
* Invalid data
* Noise

So, we use various filtering techniques to clean the data.

It is mandatory to detect and remove the above issues because it can negatively affect the quality of the outcome.

## Data Analysis

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

* **Selection of analytical techniques**
* **Building models**
* **Review the result**

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as **Classification, Regression, Cluster analysis, Association,** etc. then build the model using prepared data, and evaluate the model.

## Train Model

Now the next step is to train the model, in this step we train our model to improve its performance for better outcome of the problem.

We use datasets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and, features.

## Test Model

Once our machine learning model has been trained on a given dataset, then we test the model. In this step, we check for the accuracy of our model by providing a test dataset to it.

Testing the model determines the percentage accuracy of the model as per the requirement of project or problem.

## Deployment

The last step of machine learning life cycle is deployment, where we deploy the model in the real-world system.

If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is similar to making the final report for a project.

**3.8 DESIGN (UI/UX AND BACKEND)**

**UI/UX**

The designing of this project has been done using Streamlit app , Streamlit’s [open-source](https://github.com/streamlit/streamlit) app framework is a breeze to get started with. It’s just a matter of:

* pip install streamlit
* streamlit hello

Streamlit is an open-source python framework for building web apps for Machine Learning and Data Science. We can instantly develop web apps and deploy them easily using Streamlit. Streamlit allows you to write an app the same way you write a python code. Streamlit makes it seamless to work on the interactive loop of coding and viewing results in the web app.

For building the ML app, we need a machine learning model. We will use this simple data to create a classification model. This is a simple logistic regression model to predict binary outcomes. The Python file in this section is named app1.py .

Streamlit is making life easy because it combines the back-end and the front-end of the app. I’ll give an example of how Streamlit replaces HTML.

In HTML, to make a numeric input bar, you have to write,

<input type="number", name="", placeholder=""/>

Fetching the value of this input can be done through a POST request, sent to the back-end script. In Streamlit, this can be done in one file by simply writing, x = st.number\_input("placeholder") and the input value is stored directly in the variable x.

Running app.py will give us this page.

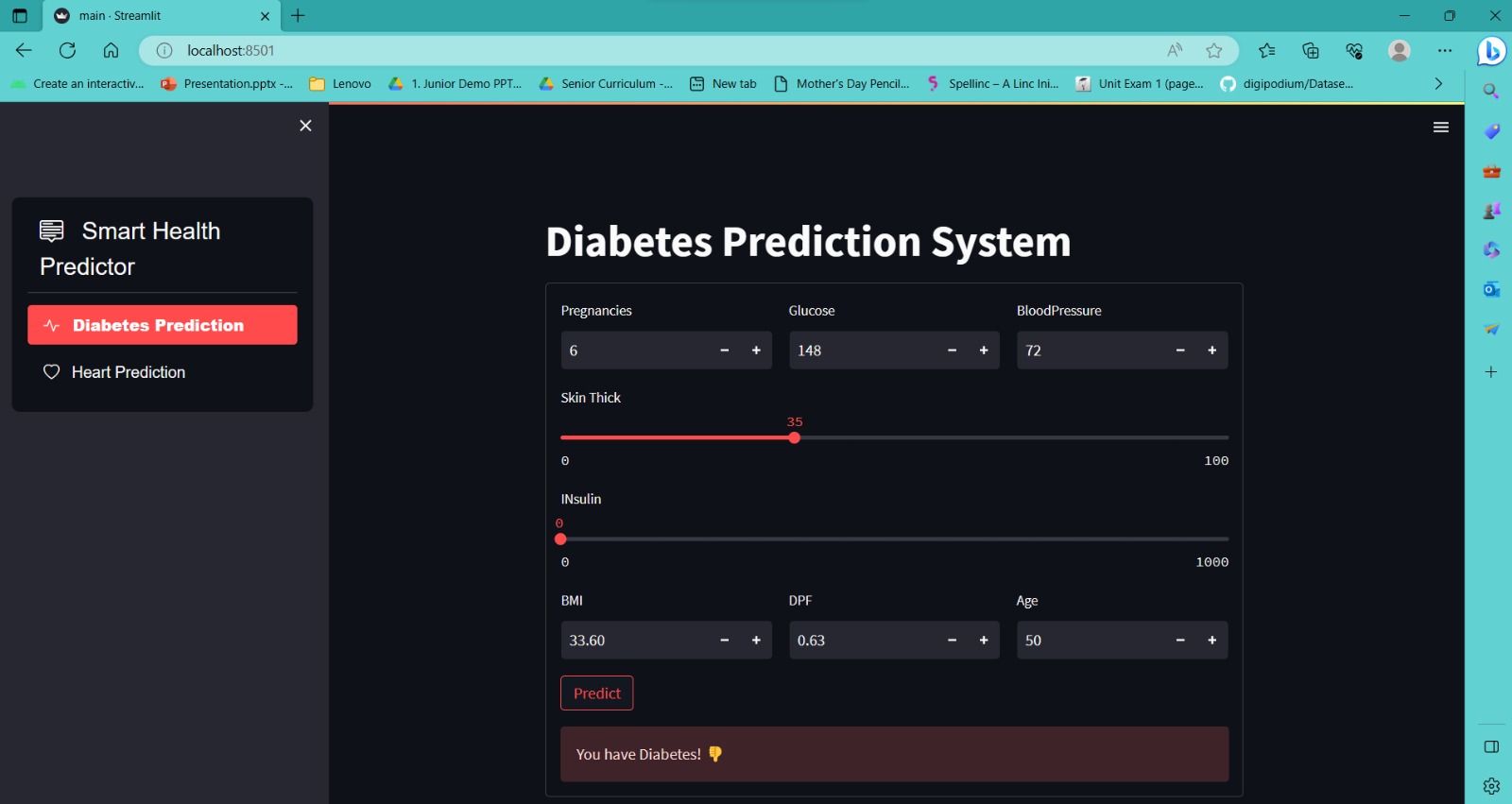


Figure 1.8

**Running the App**

To run the app we just created, go to your terminal or command prompt. You have to set the directory to your project directory. In my case, the folder name is Streamlit-ML-App. So the command I should write in my terminal is,

cd path/to/my/directory/Streamlit-ML-App

If you don’t know the path to your directory, run this code in model.py

import os  
print(os.path.dirname(os.path.abspath(\_\_file\_\_)))

The output of this code should be path/to/your/directory/Streamlit-ML-App or whatever folder name you are using. So copy the output to your terminal. Don’t forget to add cd prior.

Then you should run your Python file app.py using the following commands.

streamlit run app.py

Machine learning backends process the datasets generated from the indicators and targets calculated by the Analytics API. They are used for machine learning training, prediction and models evaluation. May be good that you also read [Analytics API](https://docs.moodle.org/dev/Analytics_API) to read some concept definitions, how these concepts are implemented in Moodle and how machine learning backend plugins fit into the analytics API.

The communication between machine learning backends and Moodle is through files because the code that will process the dataset can be written in PHP, in Python, in other languages or even use cloud services. This needs to be scalable so they are expected to be able to manage big files and train algorithms reading input files in batches if necessary.

Machine learning backend is a new Moodle plugin type. They are stored in lib/mlbackend, where you can add your own plugins.

**3.9 CODING**

**DATASET DETAILS OF HEART DISEASE PREDICTION MODEL**

• Of the 76 attributes available in the dataset,14 attributes are considered for the prediction of the output.

• Heart Disease UCI : <https://archive.ics.uci.edu/ml/datasets/Heart+Disease>

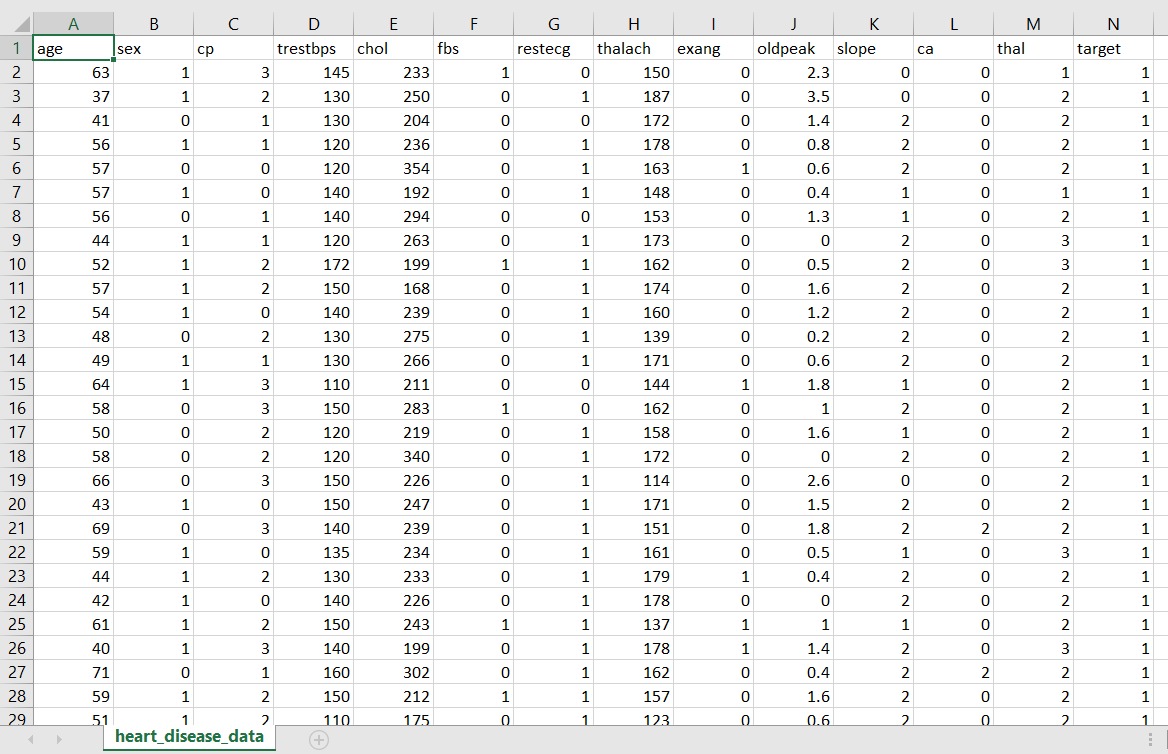


Figure 1.9

**Input dataset attributes**

● Gender (value 1: Male; value 0 : Female)

● Chest Pain Type (value 1: typical type 1 angina, value 2: typical type angina, value 3: non-angina pain; value 4: asymptomatic)

● Fasting Blood Sugar (value 1: > 120 mg/dl; value 0:< 120 mg/dl)

● Exang – exercise induced angina (value 1: yes; value 0: no)

● CA – number of major vessels colored by fluoroscopy (value 0 – 3)

● Thal (value 3: normal; value 6: fixed defect; value 7:reversible defect)

● Trest Blood Pressure (mm Hg on admission to the hospital)

● Thalach – maximum heart rate achieved

● Age in Year

● slope

● Cholestrol

● Restecg

* Old peak
* Target

**Importing libraries**

First thing first. We have first imported the necessary libraries and datasets. In this project, we are using Numpy arrays. The read\_csv method from the Pandas library enables us to read the \*.csv (comma-separated value) file format heart disease dataset published by UCI into the dataframe. The DataFrame object is the primary Pandas data structure which is a two-dimensional table with labelled axes – along rows and along with columns. Various data manipulation operations can be applied to the Pandas dataframe along rows and columns.

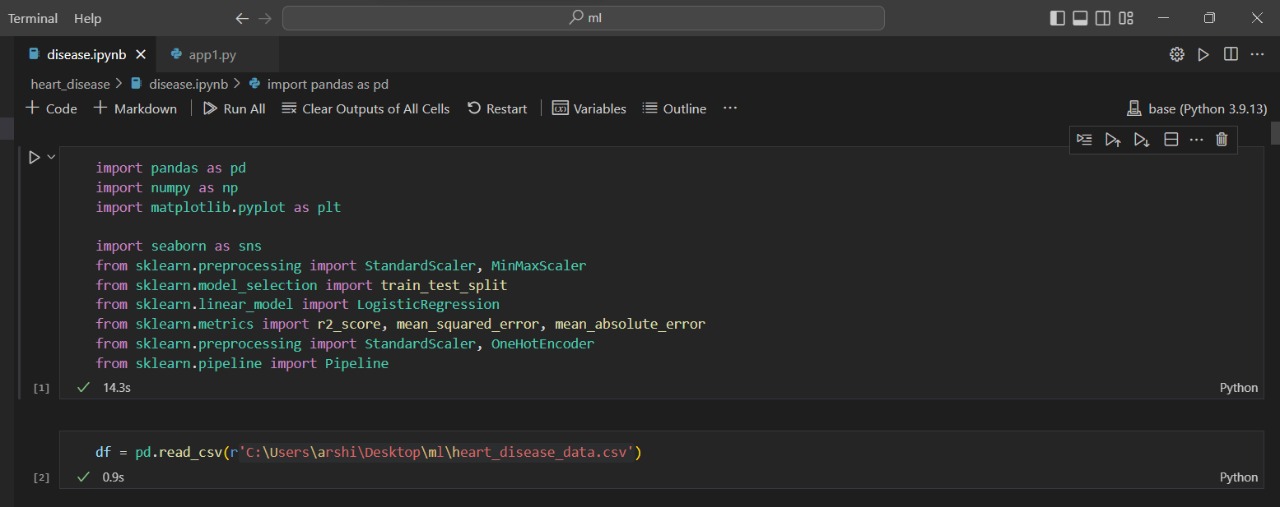


Figure 1.10

We use the scikit-learn (sklearn) library for our machine learning studies. The scikit-learn library is an open-source Python library for predictive data analysis and machine learning and is built on top of Numpy, SciPy and Matplotlib. The SciPy ecosystem is used for scientific computing and provides optimized modules for Linear Algebra, Calculus, ODE solvers and Fast Fourier transforms among others. The sklearn preprocessing module implements function like scaling, normalizing and binarizing data. The StandardScaler standardizes the features by making the mean equal to zero and variance equal to one. The fit\_transform() method achieves the dual purpose of (i) the fit() method by fitting a scaling algorithm and finding out the parameters for scaling (ii) the transform method, where the actual scaling transformation is applied by using the parameters found in the fit() method. Many machine learning algorithms are designed based on the assumption of expecting normalized/scaled data and standard scaling is thus one of the methods that help in improving the accuracy of machine learning models.

**Data Exploration**

****

Figure 1.11

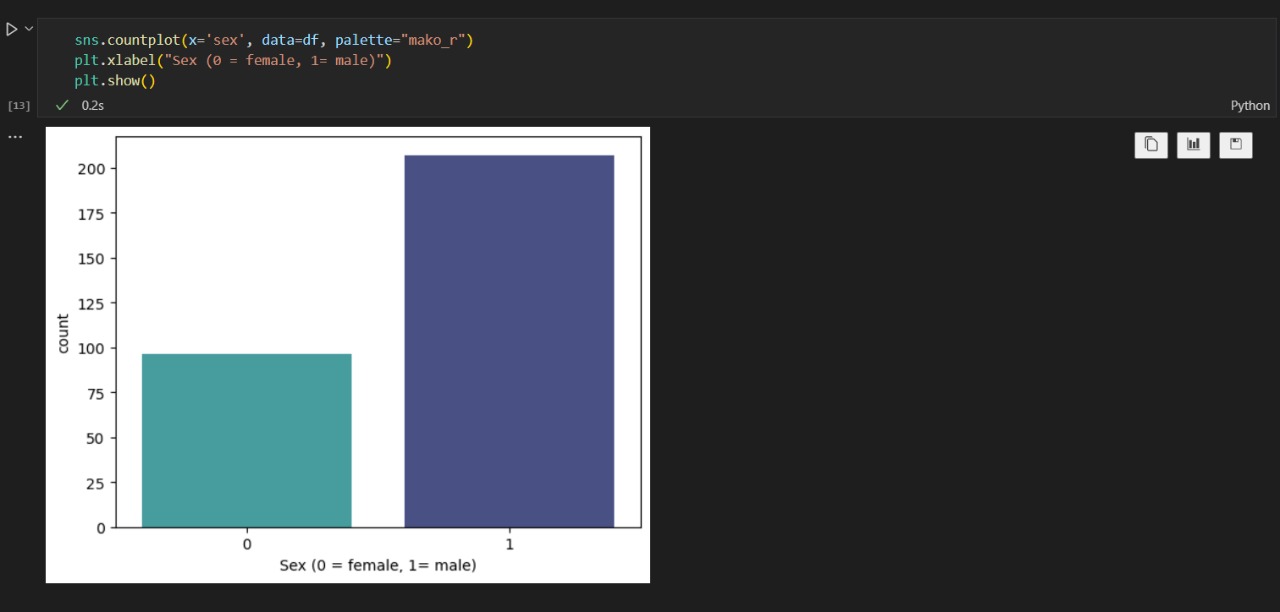


Figure 1.12

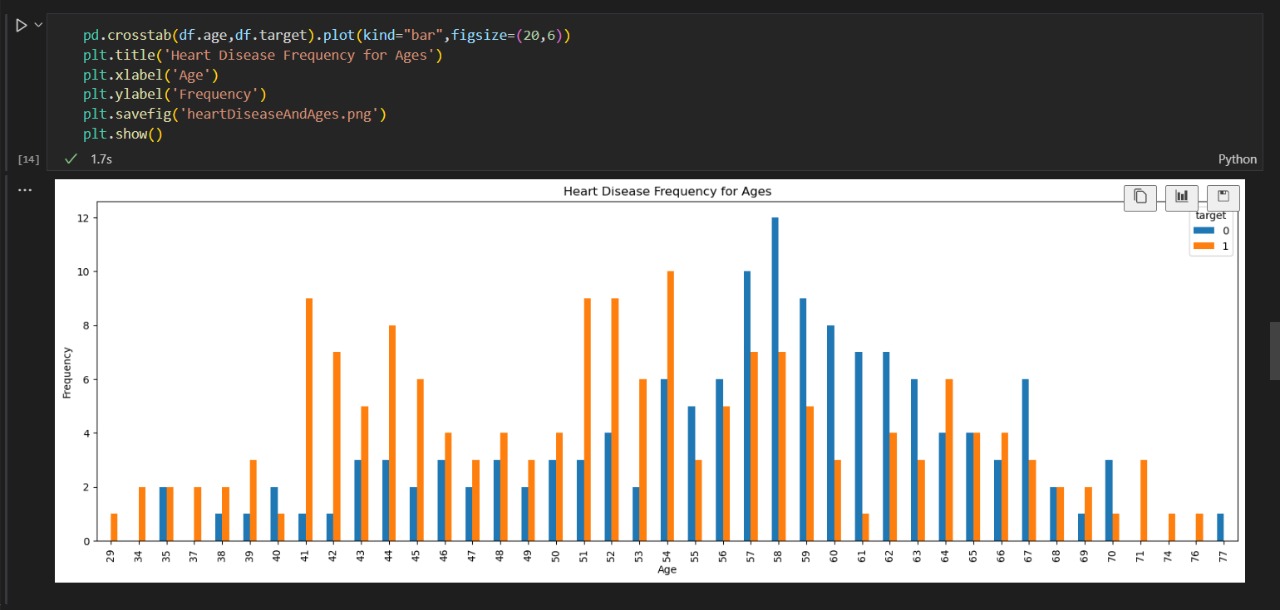


Figure 1.13

**Seaborn heatmap**

The Seaborn heatmap API provides the colour encoded plot for 2-D matrix data. The Pandas dataframe corr() method provides pairwise correlation (movement of the two variables in relation to each other) of columns in the dataframe. NA or null values are excluded by this method. The method allows us to find positive and negative correlations and strong and weak correlations between the various columns and the target variable. This can help us in feature selection. Weakly correlated features can be neglected. Positive and negative correlations can be used to describe model predictions. Positive correlation implies that as the value of one variable goes up, the value of the other variable also goes up. A negative correlation implies that as the value of one variable goes down, the value of the other variable also goes down. Zero correlation implies that there is no linear relationship between the variables. linewidth gives the width of the line that divides each cell in the heatmap. Setting can not to True, labels each cell with the corresponding correlation value. cmap value defines the mapping of the data value to the colorspace.

****

Figure 1.14

**Machine Learning Pipeline**

A Machine Learning pipeline is a process of automating the workflow of a complete machine learning task. It can be done by enabling a sequence of data to be transformed and correlated together in a model that can be analyzed to get the output. A typical pipeline includes raw data input, features, outputs, model parameters, ML models, and Predictions. Moreover, an ML Pipeline contains multiple sequential steps that perform everything ranging from data extraction and pre-processing to model training and deployment in Machine learning in a modular approach. It means that in the pipeline, each step is designed as an independent module, and all these modules are tied together to get the final result.

The ML pipeline is a high-level API for MLlib within the "spark.ml" package. A typical pipeline contains various stages.



Figure 1.15

**Train-Test Split**

Train test split is a model validation procedure that reveals how your model performs on new data. The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

It is a fast and easy procedure to perform, the results of which allow you to compare the performance of machine learning algorithms for your predictive modeling problem. Although simple to use and interpret, there are times when the procedure should not be used, such as when you have a small dataset and situations where additional configuration is required, such as when it is used for classification and the dataset is not balanced.

We will split our data. 80% of our data will be train data and 20% of it will be test data.

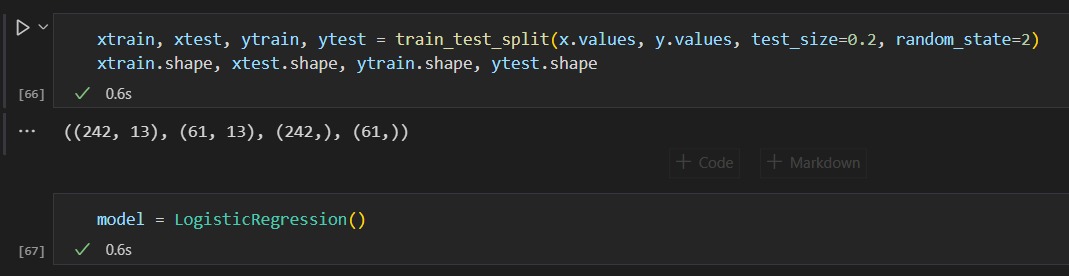


Figure 1.16

**Model Fitting and Prediction**

The sklearn linear\_model class implements a variety of linear models like Linear regression, Logistic regression, Ridge regression, Lasso regression etc. We import the LogisticRegression class for our classification studies. A LogisticRegression object is instantiated.

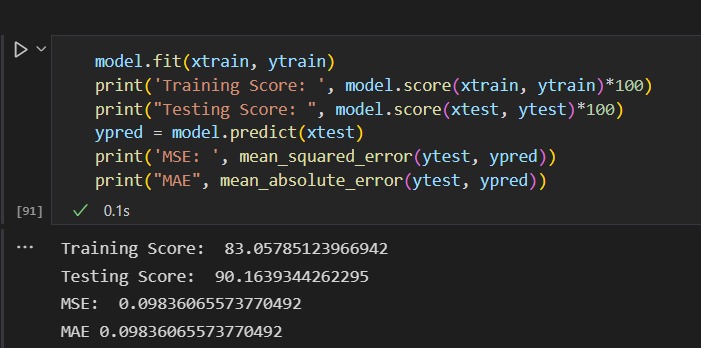
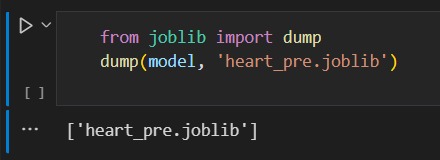


Figure 1.17

R-squared is a statistical measure that represents the goodness of fit of a regression model. The ideal value for r-square is 1. The closer the value of r-square to 1, the better is the model fitted. The Mean Squared Error (MSE) is perhaps the simplest and most common loss function, often taught in introductory Machine Learning courses. To calculate the MSE, you take the difference between your model's predictions and the ground truth, square it, and average it out across the whole dataset. Mean Absolute Error (MAE) is calculated by taking the summation of the absolute difference between the actual and calculated values of each observation over the entire array and then dividing the sum obtained by the number of observations in the array.

**Saving a Machine Learning Model**

In machine learning, while working with [scikit learn](https://www.geeksforgeeks.org/learning-model-building-scikit-learn-python-machine-learning-library/)library, we need to save the trained models in a file and restore them in order to reuse them to compare the model with other models, and to test the model on new data. The saving of data is called Serialization, while restoring the data is called Deserialization. Joblib is the replacement of pickle as it is more efficient on objects that carry large numpy arrays. These functions also accept file-like object instead of filenames. Joblib is a set of tools to provide lightweight pipelining in Python. In particular: transparent disk-caching of functions and lazy re-evaluation (memoize pattern) easy simple parallel computing.

Figure 1.18

**DATA SET DETAILS OF DIABETES PREDICTION MODEL:**

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.

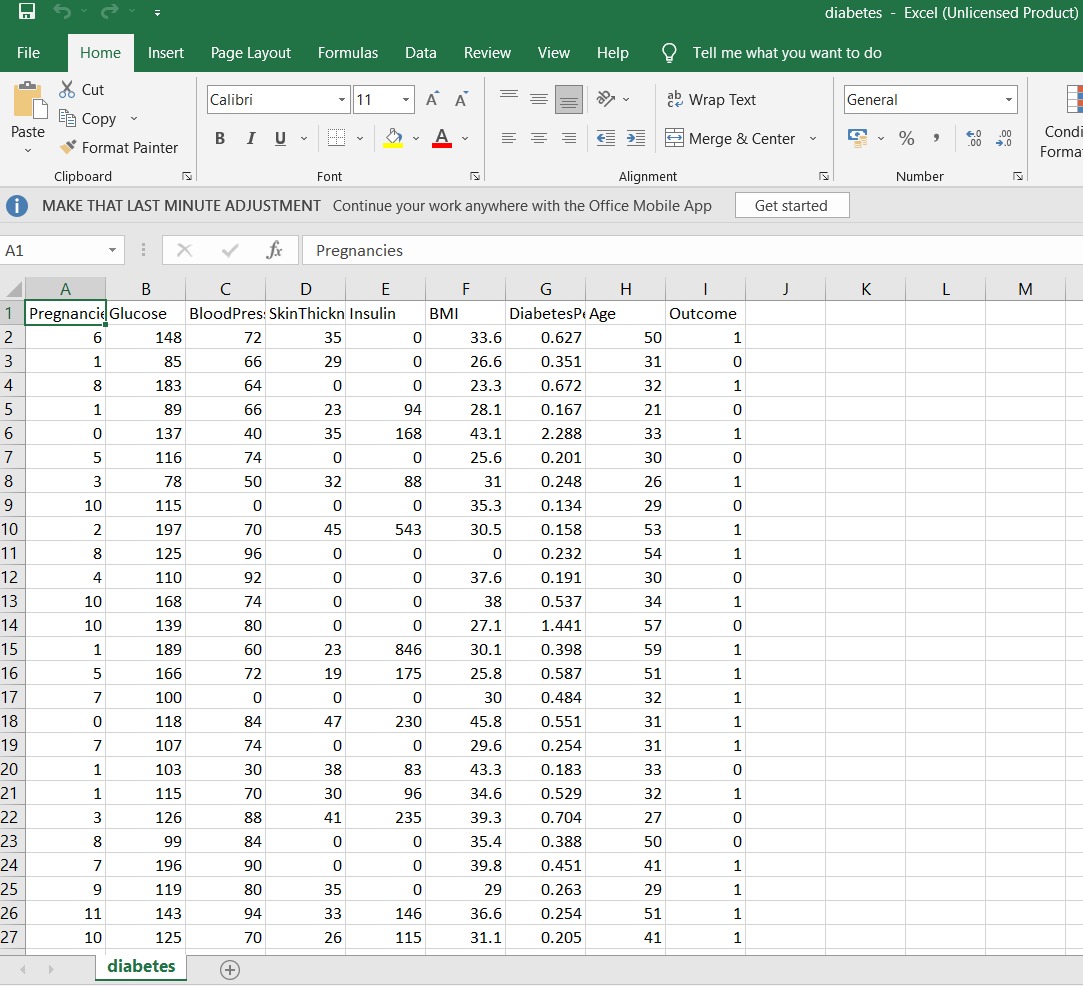


Figure 1.19

**Input dataset attributes**

* Pregnancies :- Number of times a woman has been pregnant
* Glucose :- Plasma Glucose concentration of 2 hours in an oral glucose tolerance test
* BloodPressure :- Diastollic Blood Pressure (mm hg)
* SkinThickness :- Triceps skin fold thickness(mm)
* Insulin :- 2 hour serum insulin(mu U/ml)
* BMI :- Body Mass Index ((weight in kg/height in m)^2)
* Age :- Age(years)
* DiabetesPedigreeFunction :-scores likelihood of diabetes based on family history)
* Outcome :- 0(doesn't have diabetes) or 1 (has diabetes)

**Heat map:**

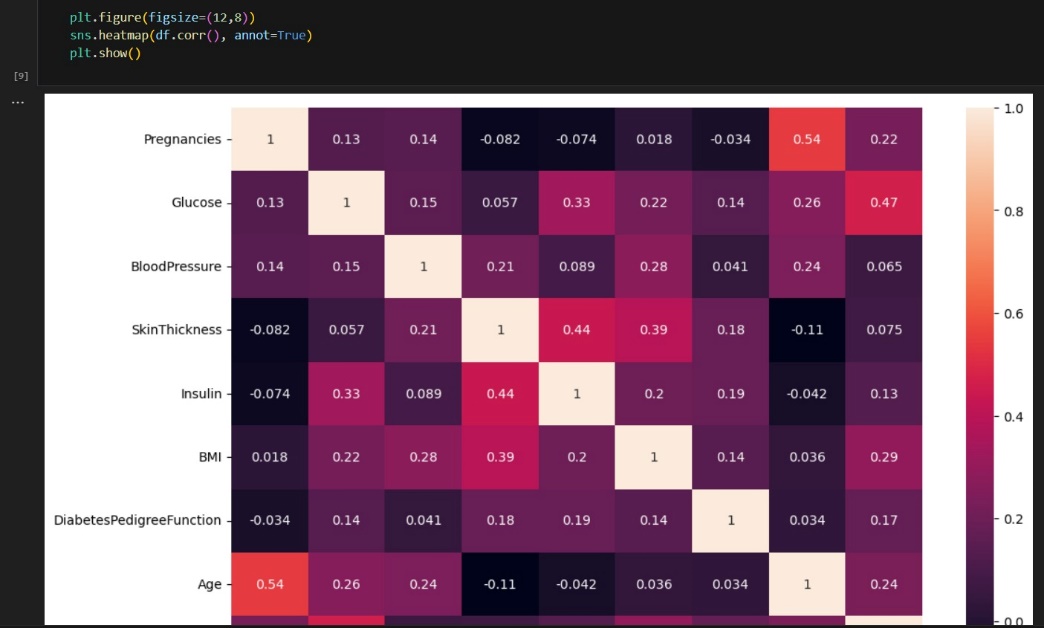
****

Figure 1.20

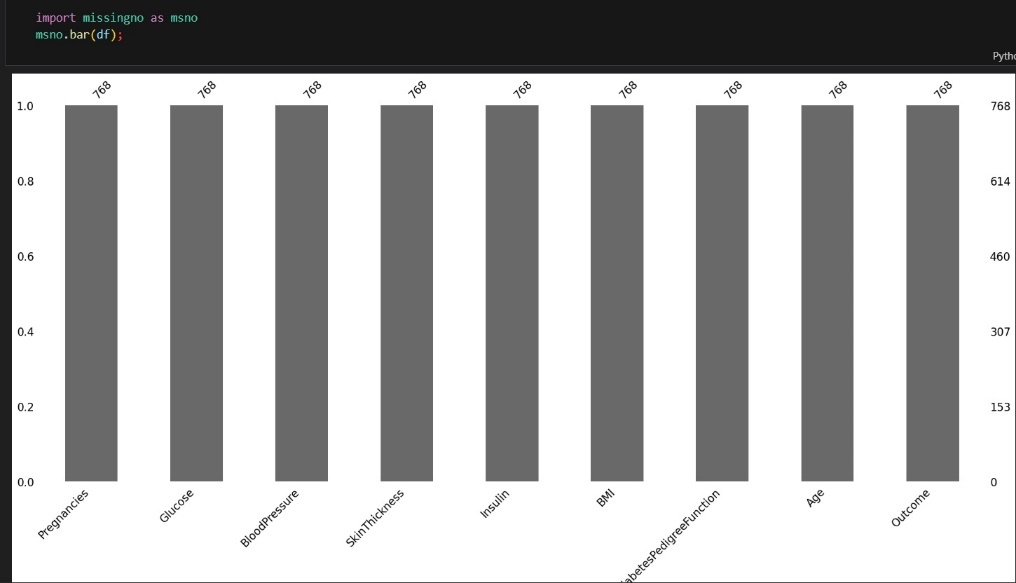
It is easy to see that there is no single feature that has a very high correlation with our outcome value.

Some of the features have a negative correlation with the outcome value and some have positive.

**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.

****Figure 1.21

**Confusion Matrix:**

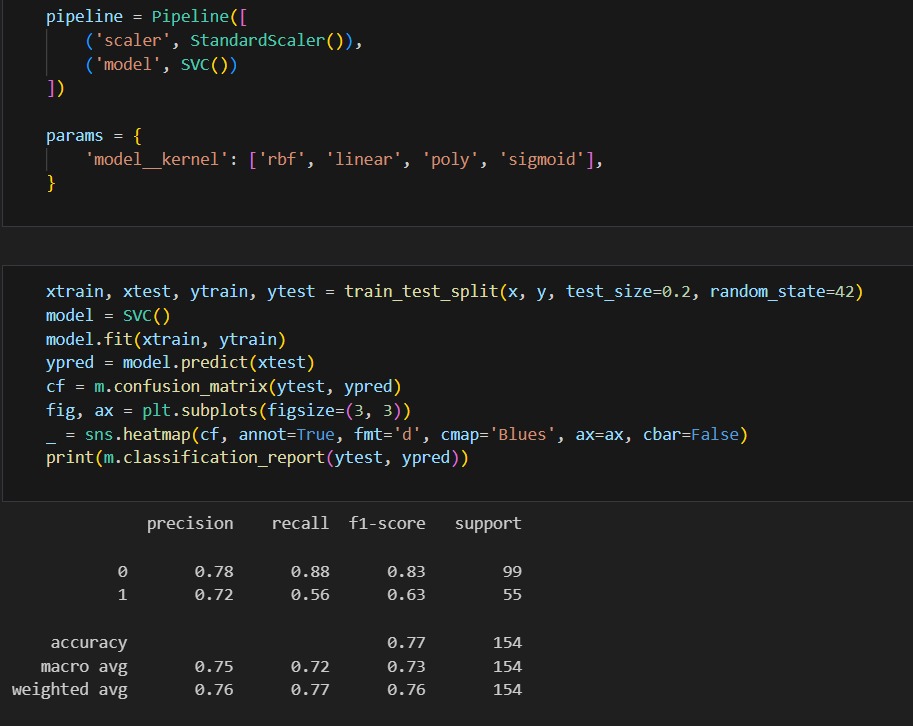
****

Figure 1.22

**Frontend : Streamlit**

 It is an open-source python library for building fast and beautiful web apps. This can be used to deploy Machine Learning models or any python projects without worrying about the frontend. Streamlit is user friendly, and no prior knowledge of other frameworks is needed. It is not necessary to create a backend, define different routes, or handle HTTP requests. Streamlit handles everything. Let’s see how easily we can deploy and perform interactive machine learning models using Streamlit.

Installation: pip install streamlit (Make sure you have Python installed in your system) Creating a python file called “app1.py” .Streamlit works by reading code directly out of a public GitHub repository. So to make the application work, we need to have a repo with the following.The .JSON file that is our model.A requirements.txt file that specifies the Python packages Streamlit needs to install for the app to run.A .py file that creates and defines our Streamlit app.Also, since the model was originally trained using a pandas DataFrame, it insists on getting new data in a similar format. That’s why we have to feed the variables to the model’s predict method using pd.DataFrame.With our prediction function defined, we need to create the app itself, and areas for user inputs. Let’s start with the app. This gives us a very basic title and image for the landing page, and a header with instructions. We have used select box , this creates a number of numerical input and dropdown menu fields that align with our model inputs. You define the range and minimum values of the numerical inputs, and the options for the categorical inputs.

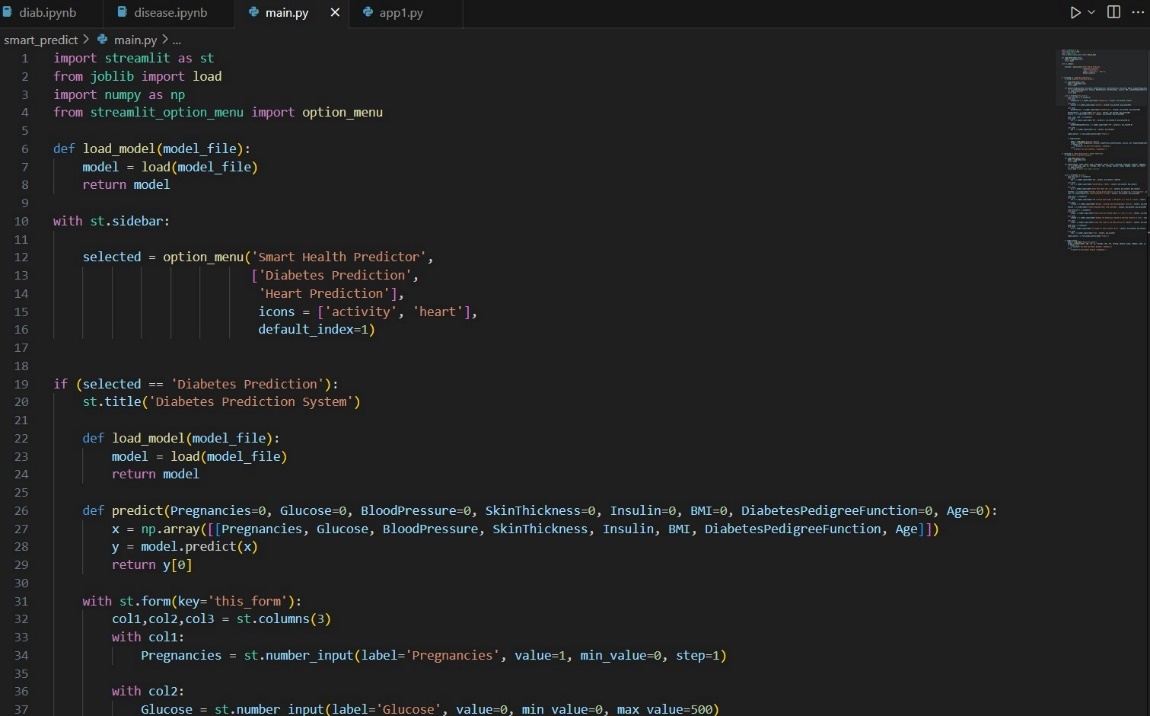


Figure 1.23

To add a background image to your Streamlit app, we have used st. markdown() with CSS.

**Load Model:** getting the ML model and create a pickle file for your model. After creating the pickle file, import all the necessary libraries.

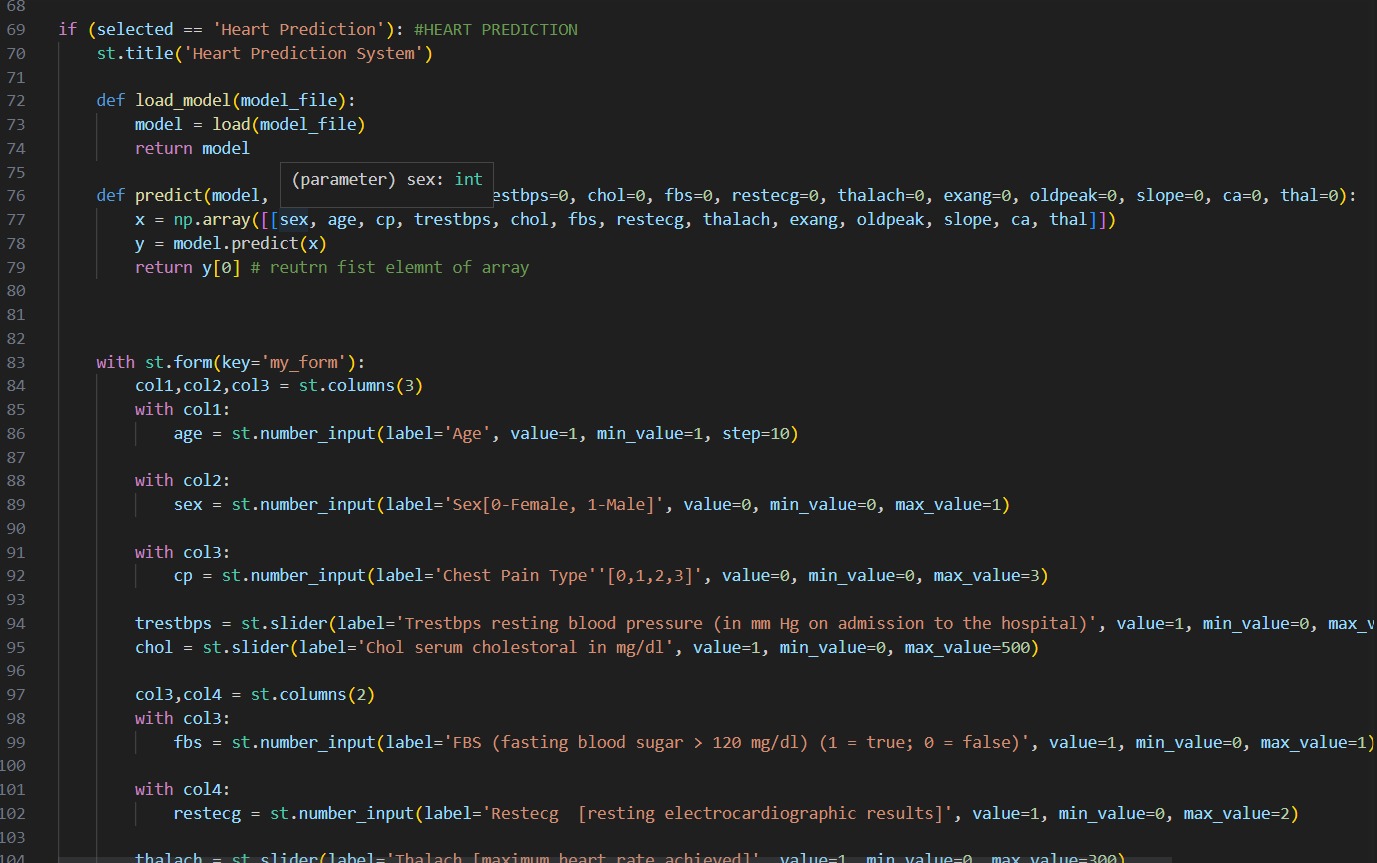
****

Figure 1.24

**3.10 TESTING**

Machine Learning testing helps spot problems in models that regular evaluation metrics might miss. These problems can come from the code that enables each element of the ML system to function or from the data where outliers and an uneven distribution split, among other things, can affect the model’s performance. It can also help mitigate future issues with the model during deployment and provide a certain level of quality assurance for the model.

ML models are frequently deployed to production, and while evaluation metrics are used to tell the performance of the dataset, model testing focuses on checking the expected behavior of the model. It is needed because there can be unexpected events in production. When models are deployed, they can encounter challenges not previously seen, and testing enables your model to perform better in varying conditions:

* Adversarial attacks: Testing models can help detect possible [adversarial attacks](https://viso.ai/deep-learning/adversarial-machine-learning/#:~:text=An%20adversarial%20attack%20is%20a,valid%20input%20to%20a%20human.). Rather than letting this attack happen in a production environment, a model can be tested with adversarial examples to increase its robustness prior to deployment.
* Data integrity and bias: Data collected from most sources are usually unstructured and might reflect human bias that can be modeled during training.  This bias might be against a particular group either by gender, race, religion, or sexuality with varying consequences in society depending on the scale of use. During the evaluation, bias can be missed because it focuses mostly on performance and not the behavior of the model given the role of the data in this case.
* Spot failure modes: Failure modes can occur when trying to deploy ML systems into production. These can be due to [performance bias failures, robustness failures or model input failures](https://www.robustintelligence.com/blog-posts/failure-modes-when-productionizing-ai-systems). Some of these failures can be missed by evaluation metrics although they can signal problems. A model with an accuracy of 90% means that the model is finding it difficult to generalize with the 10% of the data. That can prompt you to check the data and look for errors giving you better insights on how to solve it. It is not all-encompassing and so structured tests for the possible scenarios that maybe encountered need to be established and help detect failure modes.

Let us make the Logistic Regression model, predicting whether a user is suffering from a heart disease or not. Splitting the dataset to train and test. 75% of data is used for training the model and 25% of it is used to test the performance of our model. Now, it is very important to perform feature scaling here because Age and Estimated Salary values lie in different ranges. If we don’t scale the features then the Estimated Salary feature will dominate the Age feature when the model finds the nearest neighbor to a data point in the data space. Finally, we are training our Logistic Regression model. After training the model, it is time to use it to do predictions on testing data.

Many existing ML model testing practices follow manual error analysis (e.g., failure mode classification), making them slow, costly, and error-prone. A proper ML model testing framework should systematize these practices.You can map software development test types to Machine Learning models by applying their logic on Machine Learning behavior:

* Unit test. Check the correctness of individual model components.
* Regression test. Check whether your model breaks and test for previously encountered bugs.
* Integration test. Check whether the different components work with each other within your machine learning pipeline.

Specific testing tasks can belong to different categories ( model evaluation, monitoring, validation) depending on your specific problem case, circumstance, and organization structure. This article focuses on tests specific to the Machine Learning modeling problem (post-train tests), so we do not cover other test types. Make sure that you integrate your machine learning model tests into your wider [Machine Learning model monitoring framework](https://deepchecks.com/ml-model-monitoring-checklist-things-you-should-look-out-for/).

**Testing Trained Models**

For code, you can write manual test cases. This is not a great option for Machine Learning models as you cannot cover all edge cases in a multi-dimensional input space.

Instead, test model performance by doing monitoring, data slicing, or [property-based testing](https://dl.acm.org/doi/10.1145/3092703.3092711) targeted at real world problems.

You can combine this with test types that examine specifically the internal behavior of your trained models (post-train tests):

**3.11 IMPLEMENTATION**

To develop ML applications, you will have to decide on the platform, the IDE and the language for development. There are several choices available. Most of these would meet your requirements easily as all of them provide the implementation of AI algorithms discussed so far.

If you are developing the ML algorithm on your own, the following aspects need to be understood carefully −

The language of your choice − this essentially is your proficiency in one of the languages supported in ML development. The IDE that you use − This would depend on your familiarity with the existing IDEs and your comfort level.

To do this, we shall first explore our dataset using Exploratory Data Analysis (EDA) and then implement logistic regression and finally interpret the odds:

1. Import required libraries

2. Load the data, visualize and explore it

3. Clean the data

4. Deal with any outliers

5. Split the data into a training set and testing set

6. Fit a logistic regression model using sklearn

7. Apply the model on the test data and make a prediction

8. Evaluate the model accuracy using the confusion matrix

9. Create the model and obtain the regression coefficients using statsmodel

10. the essential thing is, Interpret the regression coefficient in terms of the odds

The target variable is discrete in logistic regression (unlike linear regression). It is a supervised machine learning algorithm used to address classification problems. Output from a logistic regression implementation is the estimate of the probability of a particular event occurring. As Probability goes, it is always in the range of 0 to 1.

**3.12 ALGORITHM**

**Logistic regression**

It is one of the most popular Machine Learning algorithms,which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1. Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas logistic regression is used for solving the classification problems. In Logistic regression, instead of fitting a regression line, we fit an "S"shaped logistic function, which predicts two maximum values (0 or 1). The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on itsweight, etc. Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.

**Advantages**

Logistic Regression is one of the simplest machine learning algorithms and is easy to implement yet provides great training efficiency in some cases. Also due to these reasons, training a model with this algorithm doesn't require high computation power.

The predicted parameters (trained weights) give inference about the importance of each feature. The direction of association i.e. positive or negative is also given. So we can use Logistic Regression to find out the relationship between the features.

This algorithm allows models to be updated easily to reflect new data, unlike Decision Tree or Support Vector Machine. The update can be done using stochastic gradient descent. Logistic Regression outputs well-calibrated probabilities along with classification results. This is an advantage over models that only give the final classification as results. If a training example has a 95% probability for a class, and another has a 55% probability for the same class, we get an inference about which training examples are more accurate for the formulated problem.

**Disadvantages**

Logistic Regression is a statistical analysis model that attempts to predict precise probabilistic outcomes based on independent features. On high dimensional datasets, this may lead to the model being over-fit on the training set, which means overstating the accuracy of predictions on the training set and thus the model may not be able to predict accurate results on the test set. This usually happens in the case when the model is trained on little training data with lots of features. So on high dimensional datasets, Regularization techniques should be considered to avoid overfitting (but this makes the model complex). Very high regularization factors may even lead to the model being under-fit on the training data. Non linear problems can't be solved with logistic regression since it has a linear decision surface. Linearly separable data is rarely found in real world scenarios. So the transformation of non linear features is required which can be done by increasing the number of features such that the data becomes linearly separable in higher dimensions. Non-Linearly Separable Data: It is difficult to capture complex relationships using logistic regression. More powerful and complex algorithms such as Neural Networks can easily outperform this algorithm.

**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.

The advantages of support vector machines are:

* Effective in high dimensional spaces.
* Still effective in cases where number of dimensions is greater than the number of samples.
* Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
* Versatile: different [Kernel functions](https://scikit-learn.org/stable/modules/svm.html#svm-kernels) can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

The disadvantages of support vector machines include:

* If the number of features is much greater than the number of samples, avoid over-fitting in choosing [Kernel functions](https://scikit-learn.org/stable/modules/svm.html#svm-kernels) and regularization term is crucial.
* SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation (see [Scores and probabilities](https://scikit-learn.org/stable/modules/svm.html#scores-probabilities), below).

The support vector machines in scikit-learn support both dense (numpy.ndarray and convertible to that by numpy.asarray) and sparse (any scipy.sparse) sample vectors as input. However, to use an SVM to make predictions for sparse data, it must have been fit on such data. For optimal performance, use C-ordered numpy.ndarray (dense) or scipy.sparse.csr\_matrix (sparse) with dtype=float64.

**3.13 MAINTENANCE**

Machine learning projects are highly iterative; as you progress through the ML lifecycle, you’ll find yourself iterating on a section until reaching a satisfactory level of performance, then proceeding forward to the next task (which may be circling back to an even earlier step). Moreover, a project isn’t complete after you ship the first version; you get feedback from real-world interactions and redefine the goals for the next iteration of deployment.

1. [Planning and project setup](https://www.jeremyjordan.me/ml-projects-guide/#planning)
   * Define the task and scope out requirements
   * Determine project feasibility
   * Discuss general model tradeoffs (accuracy vs speed)
   * Set up project codebase
2. [Data collection and labeling](https://www.jeremyjordan.me/ml-projects-guide/#data)
   * Define ground truth (create labeling documentation)
   * Build data ingestion pipeline
   * Validate quality of data
   * Label data and ensure ground truth is well-definend
   * Revisit Step 1 and ensure data is sufficient for the task
3. [Model exploration](https://www.jeremyjordan.me/ml-projects-guide/#exploration)
   * Establish baselines for model performance
   * Start with a simple model using initial data pipeline
   * Overfit simple model to training data
   * Stay nimble and try many parallel (isolated) ideas during early stages
   * Find SoTA model for your problem domain (if available) and reproduce results, then apply to your dataset as a second baseline
   * Revisit Step 1 and ensure feasibility
   * Revisit Step 2 and ensure data quality is sufficient
4. [Model refinement](https://www.jeremyjordan.me/ml-projects-guide/#refinement)
   * Perform model-specific optimizations (ie. hyperparameter tuning)
   * Iteratively debug model as complexity is added
   * Perform error analysis to uncover common failure modes
   * Revisit Step 2 for targeted data collection and labeling of observed failure modes
5. [Testing and evaluation](https://www.jeremyjordan.me/ml-projects-guide/#testing)
   * Evaluate model on test distribution; understand differences between train and test set distributions (how is “data in the wild” different than what you trained on)
   * Revisit model evaluation metric; ensure that this metric drives desirable downstream user behavior
   * Write tests for:
     + Input data pipeline
     + Model inference functionality
     + Model inference performance on validation data
     + Explicit scenarios expected in production (model is evaluated on a curated set of observations)
6. [Model deployment](https://www.jeremyjordan.me/ml-projects-guide/#deployment)
   * Expose model via a REST API
   * Deploy new model to small subset of users to ensure everything goes smoothly, then roll out to all users
   * Maintain the ability to roll back model to previous versions
   * Monitor live data and model prediction distributions
7. [Ongoing model maintenance](https://www.jeremyjordan.me/ml-projects-guide/#maintenance)
   * Understand that changes can affect the system in unexpected ways
   * Periodically retrain model to prevent model staleness
   * If there is a transfer in model ownership, educate the new team

**CHAPTER-4**

**RESULT ANALYSIS AND FUTURE WORK**

In this project, machine learning algorithm Logistic Regression is used to predict heart disease. Heart Disease UCI dataset, has a total of 76 attributes, out of those only 14 attributes are considered for the prediction of heart disease. Various attributes of the patient like gender, chest pain type, fasting blood pressure, serum cholesterol, exang, etc are considered for this project. The accuracy for individual algorithms has to measure and whichever algorithm is giving the best accuracy,that is considered for the heart disease prediction. For evaluating the experiment, various evaluation metrics like accuracy, confusion matrix, precision, recall, and f1-score are considered. Accuracy- Accuracy is the ratio of the number of correct predictions to the total number of inputs in the dataset. It is expressed as: Accuracy = (TP + TN) /(TP+FP+FN+TN).

Also 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 SVM algorithm.

**4.1 SCREEN –SHOT**

**TITLE: “SMART-HEALTH PREDICTOR”**

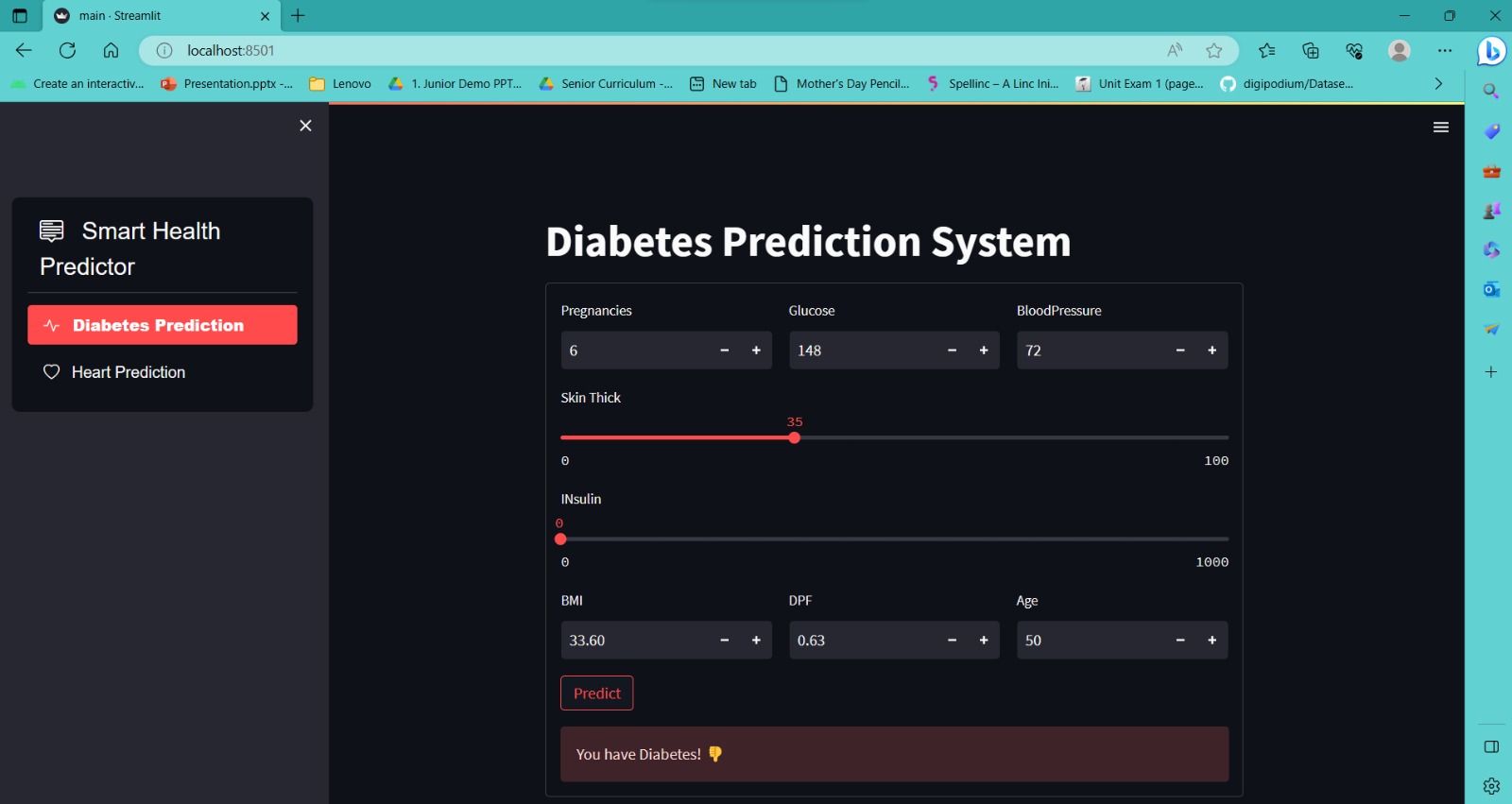
****

Figure 1.25

**OUTPUT 1:**

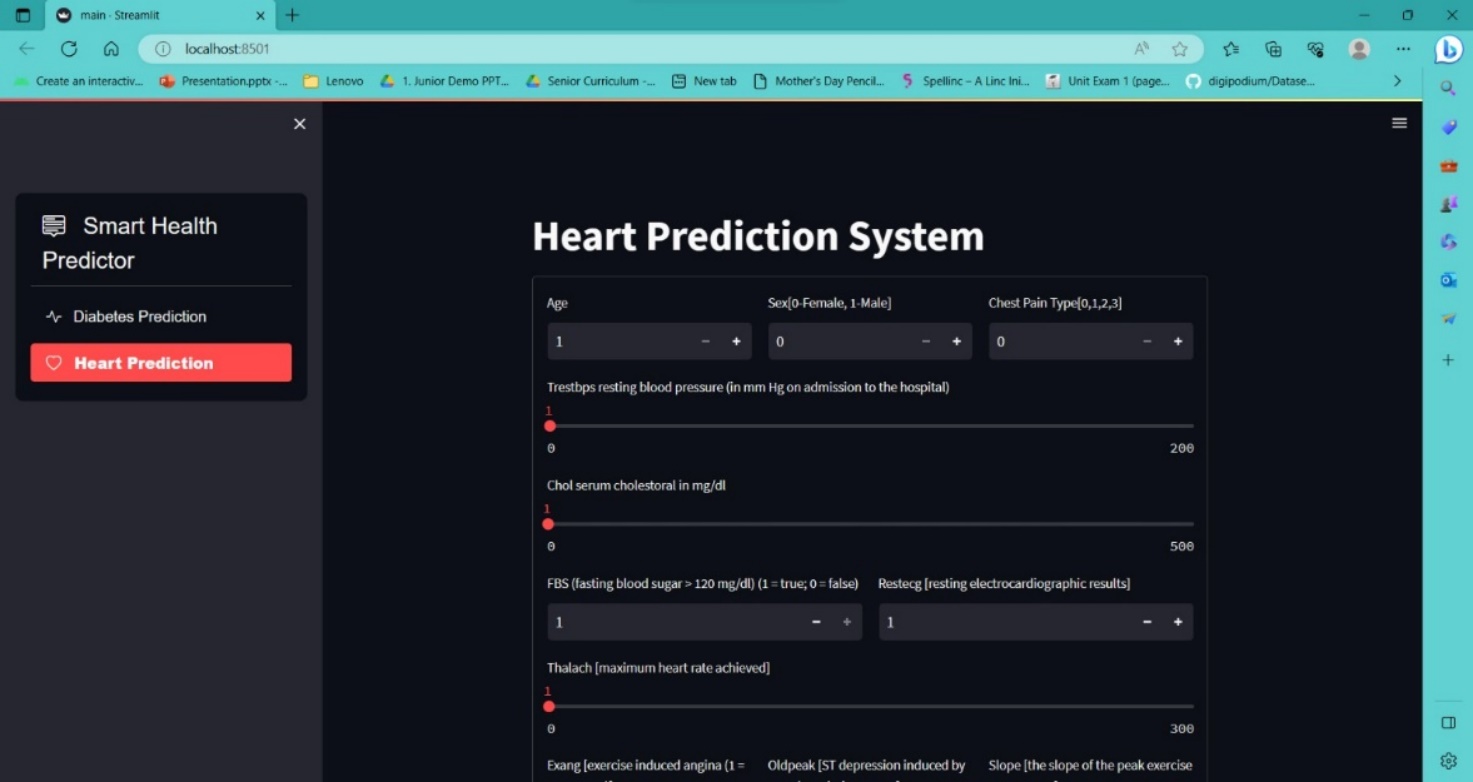
****

Figure 1.26

**OUTPUT 2:**

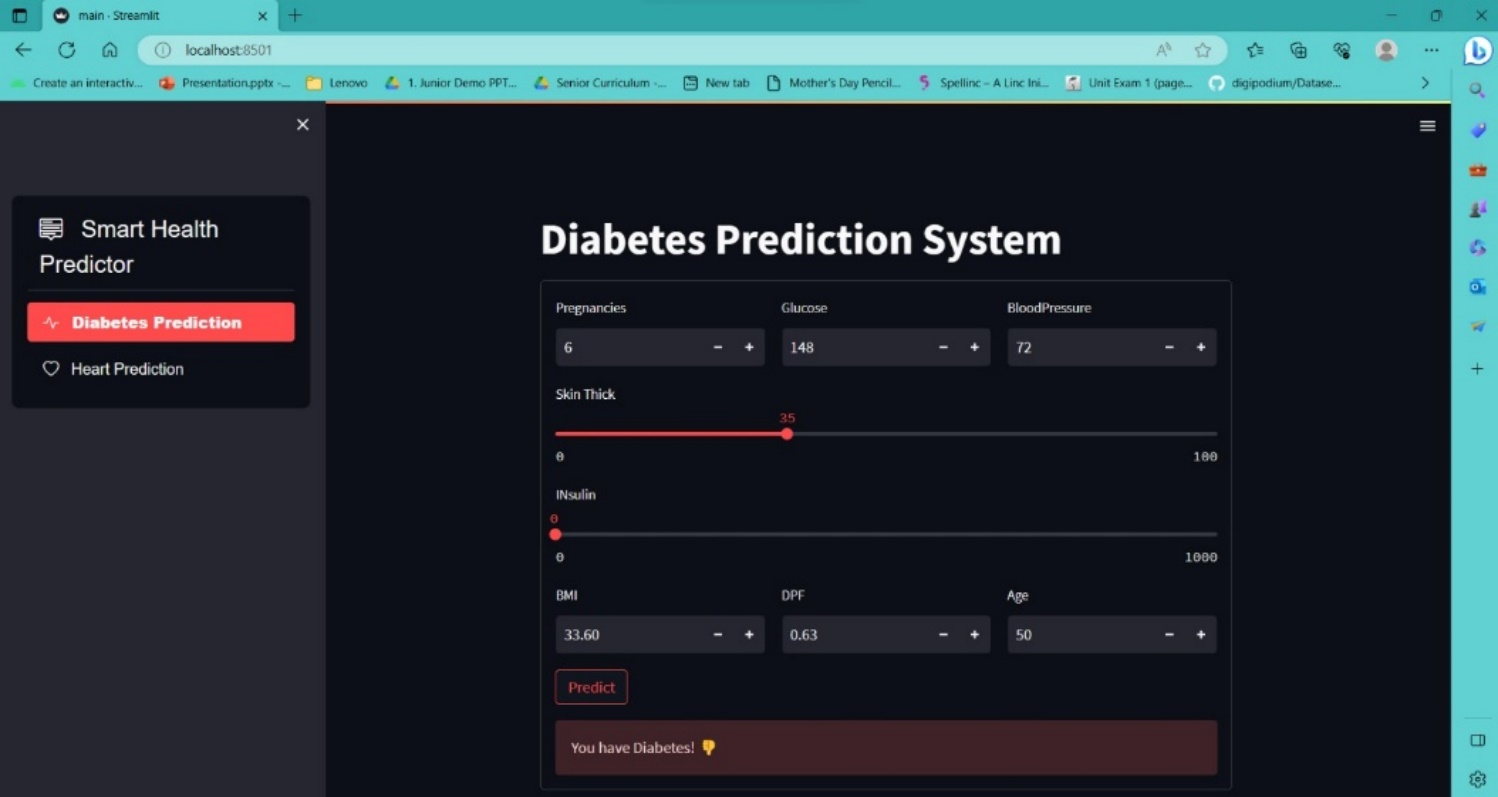
****

Figure 1.27

**4.2 RESULT**

Firstly we used some libraries provided by Python to implement this project. After creating model, the algorithm of Logistic Regression gives us the test accuracy, which is 90.1639%. Though we get a good result of 88.52% accuracy, that is not enough because it cannot guarantee that no wrong diagnosis happens. To improve accuracy, we hope to require more dataset. There are many possible improvements that could be explored to improve the scalability and accuracy of this prediction system. In the future, to predict disease we want to try different diseases such as lung cancer and want to apply more models to this project to gain more accuracy.

Training Score: 83.05

Testing Score: 90.1

Secondly for diabetes , the algorithm of SVM gives us the test accuracy, which is 75%. Though we get a good result of 88.52% accuracy, that is not enough because it cannot guarantee that no wrong diagnosis happens. To improve accuracy, we hope to require more dataset. There are many possible improvements that could be explored to improve the scalability and accuracy of this prediction system. In the future, to predict disease we want to try different diseases such as lung cancer and want to apply more models to this project to gain more accuracy.

Training Score: 70.0

Testing Score: 76.0

**CHAPTER-5**

**CONCLUSION**

Heart diseases are a major killer in India and throughout the world, application of promising technology like machine learning to the initial prediction of heart diseases will have a profound impact on society. The early prognosis of heart disease can aid in making decisions on lifestyle changes in high-risk patients and in turn reduce the complications, which can be a great milestone in the field of medicine. The number of people facing heart diseases is on a raise each year..

In this project, the algorithms used to measure the is Logistic Regression applied on the dataset. The expected attributes leading to heart disease in patients are available in the dataset which contains 76 features and 14 important features that are useful to evaluate the system. After creating model, the algorithm of Logistic Regression gives us the Training Score: 83.05 , Testing Score: 90.1. Though we get a good result of 88.52% accuracy, that is not enough because it cannot guarantee that no wrong diagnosis happens. To improve accuracy, we hope to require more dataset. There are many possible improvements that could be explored to improve the scalability and accuracy of this prediction system.

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 Diabetes Database. Experimental results determine the adequacy of the designed system with an achieved accuracy of 75% using SVM.

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.

**LIST OF FIGURE**

|  |  |  |
| --- | --- | --- |
| FIGURE NUMBER | DESCRIPTION | PAGE NUMBER |
| 1.1  1.2  1.3  1.4  1.5  1.6  1.7  1.8  1.9  1.10  1.11  1.12  1.13  1.14  1.15  1.16  1.17  1.18  1.19  1.20  1.21  1.22  1.23  1.24  1.25  1.26  1.27 | System Architecture  Selection of attribute  Collection of dataset  Data Preprocessing  Balancing of Data  Prediction of Disease  SD Model  UI/UX  Dataset 1  Coding Window  Data Exploration  Sns count  Frequency  Heatmap  Pipeline  Train split  Model Fitting  Saving the ML model  Dataset 2  Heatmap  Bar Graph  Confusion matrix  Homepage code  Homepage code  Screenshot of home page  Output1  Output2 | 3  12  13  14  14  15  16  20  22  23  24  25  25  26  27  28  28  39  30  31  31  32  33  34  44  44  45 |

**REFERENCES & BIBLIOGRAPHY**

[1] Soni J, Ansari U, Sharma D & Soni S (2011). Predictive data mining for medical diagnosis: an overview of heart disease prediction. International Journal of Computer Applications, 17(8), 43-8

[2] Dangare C S & Apte S S (2012). Improved study of heart disease prediction system using data mining classification techniques. International Journal of Computer Applications, 47(10), 44-8.

[3] Ordonez C (2006). Association rule discovery with the train and test approach for heart disease prediction. IEEE Transactions on Information Technology in Biomedicine, 10(2), 334-43.

[4] Shinde R, Arjun S, Patil P & Waghmare J (2015). An intelligent heart disease prediction system using k-means clustering and Naïve Bayes algorithm. International Journal of Computer Science and Information Technologies, 6(1), 637-9.

[5] Bashir S, Qamar U & Javed M Y (2014, November). An ensemble-based decision support framework for intelligent heart disease diagnosis. In International Conference on Information Society (i-Society 2014) (pp. 259-64). IEEE. ICCRDA 2020 IOP Conf. Series: Materials Science and Engineering 1022 (2021) 012072 IOP Publishing doi:10.1088/1757-899X/1022/1/012072 9

[6] Jee S H, Jang Y, Oh D J, Oh B H, Lee S H, Park S W & Yun Y D (2014). A coronary heart disease prediction model: the Korean Heart Study. BMJ open, 4(5), e005025.

[7] Ganna A, Magnusson P K, Pedersen N L, de Faire U, Reilly M, Ärnlöv J & Ingelsson E (2013). Multilocus genetic risk scores for coronary heart disease prediction. Arteriosclerosis, thrombosis, and vascular biology, 33(9), 2267-72.

[8] Jabbar M A, Deekshatulu B L & Chandra P (2013, March). Heart disease prediction using lazy associative classification. In 2013 International MutliConference on Automation, Computing,Communication, Control and Compressed Sensing (iMac4s) (pp. 40- 6). IEEE.

[9] Brown N, Young T, Gray D, Skene A M & Hampton J R (1997). Inpatient deaths from acute myocardial infarction, 1982-92: analysis of data in the Nottingham heart attack register. BMJ, 315(7101), 159-64.

[10] Folsom A R, Prineas R J, Kaye S A & Soler J T (1989). Body fat distribution and self-reported prevalence of hypertension, heart attack, and other heart disease in older women. International journal of epidemiologyy, 18(2), 361-7. 69

[11] Chen A H, Huang S Y, Hong P S, Cheng C H & Lin E J (2011, September). HDPS: Heart disease prediction system. In 2011 Computing in Cardiology (pp. 557- 60). IEEE.

[12] Parthiban, Latha and R Subramanian. "Intelligent heart disease prediction system using CANFIS and genetic algorithm." International Journal of Biological, Biomedical and Medical Sciences 3.3 (2008).

[13] Wolgast G, Ehrenborg C, Israelsson A, Helander J, Johansson E & Manefjord H (2016). Wireless body area network for heart attack detection [Education Corner]. IEEE antennas and propagation magazine, 58(5), 84-92.

[14] Patel S & Chauhan Y (2014). Heart attack detection and medical attention using motion sensing device -kinect. International Journal of Scientific and Research Publications, 4(1), 1-4.

[15] Piller L B, Davis B R, Cutler J A, Cushman W C, Wright J T, Williamson J D & Haywood L J (2002). Validation of heart failure events in the Antihypertensive and Lipid Lowering Treatment to Prevent Heart Attack Trial (ALLHAT) participants assigned to doxazosin and chlorthalidone. Current controlled trials in cardiovascular medicine

[16] Raihan M, Mondal S, More A, Sagor M O F, Sikder G, Majumder M A & Ghosh K (2016, December). Smartphone based ischemic heart disease (heart attack) risk prediction using clinical data and data mining approaches, a prototype design. In 2016 19th International Conference on Computer and Information Technology (ICCIT) (pp. 299-303). IEEE.