PROJECT REPORT

On

HEART DISEASE PREDICTION SYSTEM

In partial fulfillment of requirement for the degree

Of

Bachelor of Technology

In

Computer Science & Engineering

Submitted by

Dhanushri Nawghare (21100BTCSBS09690)

Veda Diwan (2100BTCSBS09717)

Under the guidance by

Prof. Sunny Bagga

SHRI VAISHNAV VIDYAPEETH VISHWAVIDYALAYA

SHRI VAISHNAV INSTITUTE OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

2021-2025

Table of Content

SHRI VAISHNAV VIDYAPEETH VISHWAVIDYALAYA

SHRI VAISHNAV INSTITUTE OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Declaration

We declared that the work which is being presented in this project **Heart Disease Prediction System** in partial fulfillment of degree of **Bachelors of Technology in Computer Science Engineering** is an authentic record of a work carried out under the supervision and guidance of **Mr. Sunny Bagga** assistant professor of Computer Science and Engineering the matter avoided in this project has not been submitted for the award of any other degree.

Dhanushri Nawghare

Veda Diwan

SHRI VAISHNAV VIDYAPEETH VISHWAVIDYALAYA

SHRI VAISHNAV INSTITUTE OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Certificate

This is the certified that **Miss Dhanushri Nawghare and Miss Veda Diwan** working in a team has satisfactorily completed the project entitled **"Heart Disease Prediction System"** under the guidance of **Mr. Sunny Bagga** in the partial fulfillment and of degree of **Bachelors of Technology in Computer Science and Engineering** awarded by Shri Vaishnav Institute of Information Technology affiliated to Shri Vaishnav Vidyapeeth Vishwavidyalaya Indore during the academic year July December 2023.

Dhanushri Nawghare

Veda Diwan

Abstract

The main aim of the **"Heart Disease Prediction System"** is to provide in advance prediction of heart disease. Cardiovascular disease is the predominant cause of that across the globe over the last digits in the developed as well as under developed and developing countries. Timely detection of cardiac diseases and continuous separation of doctors can reduce the death rate. However, it is not possible to monitor patients on the daily basis in all cases accurately and consultation of a patient for 24 hours by a doctor is not available since it requires more patients time and expertise. So to detect heart disease but I'm always days we have developed a prediction system using various machine learning algorithms like K nearest neighbor, Logistic regression, Support factor machine, Naive Bayes classifier and many more. In this project we have predicted people having cardiovascular disease by extracting the patience medical information that leads to a life-threatening heart disease from a data set that includes patient's medical history including chest pain, sugar level, blood pressure, etc. The algorithm used in building the given model are logistic regression, random forest, SVM, KNN and many more. The highest accuracy of a model is 93%.

Therefore, in conclusion this project helps us to predict the patient who are diagnosed with heart disease by pre-processing the dataset and applying **"Random Forest" algorithm with an accuracy 94%** on our model which is far better than the previous model having less accuracy.

TABLE OF FIGURES

CHAPTER -1

INTRODUCTION

1. Introduction

Heart disease, also known as cardiovascular disease (CVD), stands as a formidable global health challenge, exacting a profound toll on individuals and societies alike. With its multifaceted manifestations, heart disease encompasses a spectrum of conditions affecting the heart and blood vessels, including coronary artery disease, heart failure, arrhythmias, and valvular disorders. The World Health Organization (WHO) identifies cardiovascular diseases as the leading cause of death globally, responsible for approximately 17.9 million fatalities annually, accounting for 31% of all global deaths. The staggering prevalence of heart disease underscores the urgency of comprehensive research, public health initiatives, and individual lifestyle modifications to mitigate its impact.

Developing an effective heart disease prediction and diagnosis system is crucial for early intervention and improved patient outcomes. Such systems typically integrate advanced technologies, medical data analysis, and predictive modeling to assess the risk of heart disease, facilitate timely diagnosis, and guide personalized treatment plans.

This particular research focusses person who is more likely to have a heart disease based on various medical attributes. We also made a predictive system to predict which patient is diagnosed with disease.

Predicting Heart Disease manually is a very hard thing. Developing, a robust heart disease prediction and diagnosis system can play a pivotal role in revolutionizing cardiovascular healthcare, enabling early detection, personalized intervention, and improved patient outcomes. It represents a promising avenue for advancing preventive medicine and reducing the global burden of heart disease.

Machine learning (ML) plays a crucial role in heart disease prediction and diagnosis systems by leveraging computational algorithms to analyze complex datasets and identify patterns, trends, and relationships within the data. machine learning in heart disease prediction and diagnosis systems involves a systematic process of data preprocessing, algorithm selection, training, evaluation, and deployment. The iterative nature of the ML pipeline, along with continuous monitoring and

improvement, contributes to the system's ability to provide accurate and timely predictions, ultimately enhancing patient care and outcomes.

2. Machine Learning

Machine learning is a subset of AI, which enables the machine to automatically learn from data, improve performance from past experiences, and make predictions. Machine learning contains a set of algorithms that work on a huge amount of data. Data is fed to these algorithms to train them, and on the basis of training, they build the model & perform a specific task.

Tom Mitchell provides a more modern definition: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

2.1.Supervised Learning

Supervised machine learning is based on supervision. It means in the supervised learning technique, we train the machines using the "labelled" dataset, and based on the training, the machine predicts the output. Here, the labelled data specifies that some of the inputs are already mapped to the output.

More preciously, we can say; first, we train the machine with the input and corresponding output, and then we ask the machine to predict the output using the test dataset.

Supervised machine learning can be classified into two types of problems, which are given below:

- Classification: Classification algorithms are used to solve the classification problems in which the output variable is categorical, such as "Yes" or No, Male or Female, Red or Blue, etc.

- Regression: Regression algorithms are used to solve regression problems in which there is a linear relationship between input and output variables.

2.2. Unsupervised Learning

Unsupervised learning is different from the Supervised learning technique; as its name suggests, there is no need for supervision. It means, in unsupervised machine learning, the machine is trained using the unlabeled dataset, and the machine predicts the output without any supervision.

In unsupervised learning, the models are trained with the data that is neither classified nor labelled, and the model acts on that data without any supervision. Unsupervised Learning can be further classified into two types, which are given below:

- Clustering: The clustering technique is used when we want to find the inherent groups from the data.

- Association: Association rule learning is an unsupervised learning technique, which finds interesting relations among variables within a large dataset.

CHAPTER – 2

MACHINE LEARNING ALGORITHMS

1. Machine Learning Algorithms

Machine Learning algorithms are the programs that can learn the hidden patterns from the data, predict the output, and improve the performance from experiences on their own.

1.1. Modeling Algorithms:

I. Naïve Bayes Algorithm:

Naïve Bayes classifier is a supervised learning algorithm, which is used to make predictions based on the probability of the object. The algorithm named as Naïve Bayes as it is based on Bayes theorem, and follows the naïve assumption that says' variables are independent of each other.

II. Logistic Regression:

Logistic regression is the supervised learning algorithm, which is used to predict the categorical variables or discrete values. It can be used for the classification problems in machine learning, and the output of the logistic regression algorithm can be either Yes or NO, 0 or 1, Red or Blue, etc.

Logistic regression is similar to the linear regression except how they are used, such as Linear regression is used to solve the regression problem and predict continuous values, whereas Logistic regression is used to solve the Classification problem and used to predict the discrete values.

III. Decision Tree Algorithm

A decision tree is a supervised learning algorithm that is mainly used to solve the classification problems but can also be used for solving the regression problems. It can work with both categorical variables and continuous variables. It shows a tree-like structure that includes nodes and branches, and starts with the root node that expand on further branches till the leaf node. The internal node is used to represent the features of the dataset, branches show the decision rules, and leaf nodes represent the outcome of the problem.

IV. Support Vector Machine Algorithm

A support vector machine or SVM is a supervised learning algorithm that can also be used for classification and regression problems. However, it is primarily used for classification problems. The goal of SVM is to create a hyperplane or decision boundary that can segregate datasets into different classes. The data points that help to define the hyperplane are known as support vectors, and hence it is named as support vector machine algorithm.

V. Random Forest Algorithm

Random forest is the supervised learning algorithm that can be used for both classification and regression problems in machine learning. It is an ensemble learning technique that provides the predictions by combining the multiple classifiers and improve the performance of the model.

It contains multiple decision trees for subsets of the given dataset, and find the average to improve the predictive accuracy of the model. A random-forest should contain 64-128 trees. The greater number of trees leads to higher accuracy of the algorithm.

VI. K-Nearest Neighbour (KNN)

K-Nearest Neighbour is a supervised learning algorithm that can be used for both classification and regression problems. This algorithm works by assuming the similarities between the new data point and available data points. Based on these similarities, the new data points are put in the most similar categories. It is also known as the lazy learner algorithm as it stores all the available datasets and classifies each new case with the help of K-neighbours.

CHAPTER – 3

DESCRIPTION OF DATASET

1. **About Dataset**

   This data set dates from 1988 and consists of four databases: Cleveland, Hungary, Switzerland, and Long Beach V. It contains 76 attributes, including the predicted attribute, but all published experiments refer to using a subset of 14 of them. The "target" field refers to the presence of heart disease in the patient. It is integer valued 0 = no disease and 1 = disease.

2. **Understanding Features:**

   - **(age):** The age of the patient. It must be in years.
   - **(sex):** Displays gender of the individual.
     - Male – 0
     - Female – 1
   - **(cp):** Chest Pain Type is a categorical variable indicating the type of chest pain experienced by the patient, with four possible values.
     - Value 1: typical angina
     - Value 2: atypical angina
     - Value 3: non-anginal pain
     - Value 4: asymptomatic
   - **(trestbps):** resting blood pressure (in mm Hg on admission to the hospital)
   - **(chol):** serum cholestoral in mg/dl.
   - **(fbs):** fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
   - **(restecg):** resting electrocardiographic results
     - Value 0: normal
     - Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)

- Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria

❖ **(thalach):** maximum heart rate achieved

❖ **(exang):** exercise induced angina (1 = yes; 0 = no)

❖ **(oldpeak):** ST depression induced by exercise relative to rest .

❖ **(slope):** the slope of the peak exercise ST segment

- Value 1: upsloping
- Value 2: flat
- Value 3: downsloping

❖ **(ca):** number of major vessels (0-3) colored by flourosopy.

❖ **(thal):** A categorical variable representing the type of thalassemia,

- 0 = normal
- 1 = fixed defect
- 2 = reversable defect

❖ **(target):** the predicted attribute

- 0 = no disease
- 1 = disease.

3. Characteristics Of Dataset:

This dataset is already pre-processed after combing the four database Cleveland, Hungary, Switzerland, and Long Beach V. From the below figures we can infer that there is no null values in the dataset. Also it consist of 1025 rows and 14 columns. The multi class variable and binary classification are introduced for the attributes of the given dataset.

The data pre-processing is carried out by converting medical records into diagnosis values. Also, fig 2 represents various statical results like standard deviation, mean, etc., for the dataset.

```
In [3]: ## To check the presence of
        df.isnull().sum()

Out[3]: age       0
        sex       0
        cp        0
        trestbps  0
        chol      0
        fbs       0
        restecg   0
        thalach   0
        exang     0
        oldpeak   0
        slope     0
        ca        0
        thal      0
        target    0
        dtype: int64
```

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1025 non-null   int64
 1   sex       1025 non-null   int64
 2   cp        1025 non-null   int64
 3   trestbps  1025 non-null   int64
 4   chol      1025 non-null   int64
 5   fbs       1025 non-null   int64
 6   restecg   1025 non-null   int64
 7   thalach   1025 non-null   int64
 8   exang     1025 non-null   int64
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64
 11  ca        1025 non-null   float64
 12  thal      1025 non-null   float64
 13  target    1025 non-null   int64
dtypes: float64(3), int64(11)
memory usage: 112.2 KB
```

FIG 3.1 UNDERSTANDING FIGURES

```
[ ] df.describe()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000
mean	54.434146	0.695610	0.942439	131.611707	246.00000	0.149268	0.529756	149.114146	0.336585	1.071512	1.385366	0.754146	2.323902	0.513171
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	23.005724	0.472772	1.175053	0.617755	1.030798	0.620660	0.500070
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	132.000000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	152.000000	0.000000	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	166.000000	1.000000	1.800000	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

FIG 3.2 DESCRIBING DATASET

4. Correlation Matrix Of The Dataset

A correlation matrix is a table showing correlation coefficients between variables. Each cell in the table shows the correlation between two variables. A correlation matrix is used to summarize data, as an input into a more advanced analysis, and as a diagnostic for advanced analyses.

The Spearman Correlation coefficient is also known as Spearman's Rank Correlation coefficient or Spearman's RHO. The Spearman Correlation coefficient can range from -1.0 to +1.0.

- If the value is 1, it is said to be a positive correlation between two variables. This means that when one variable increases, the other variable also increases.
- If the value is -1, it is said to be a negative correlation between the two variables. This means that when one variable increases, the other variable decreases.
- If the value is 0, there is no correlation between the two variables. This means that the variables changes in a random manner with respect to each other.

Applications of a correlation matrix

There are three broad reasons for computing a correlation matrix:

- To summarize a large amount of data where the goal is to see patterns. In our example above, the observable pattern is that all the variables highly correlate with each other.
- To input into other analyses. For example, people commonly use correlation matrixes as inputs for exploratory factor analysis, confirmatory factor analysis, structural equation models, and linear regression when excluding missing values pairwise.
- As a diagnostic when checking other analyses. For example, with linear regression, a high amount of correlations suggests that the linear regression estimates will be unreliable.

```
## finding the corelation different attributes
plt.figure(figsize=(12,10))
p= sns.heatmap(df.corr(), vmin=-1,vmax=1,annot= True)
plt.title("Correlation amoung the all the attributes",fontsize=20)
```

```
Text(0.5, 1.0, 'Correlation amoung the all the attributes')
```

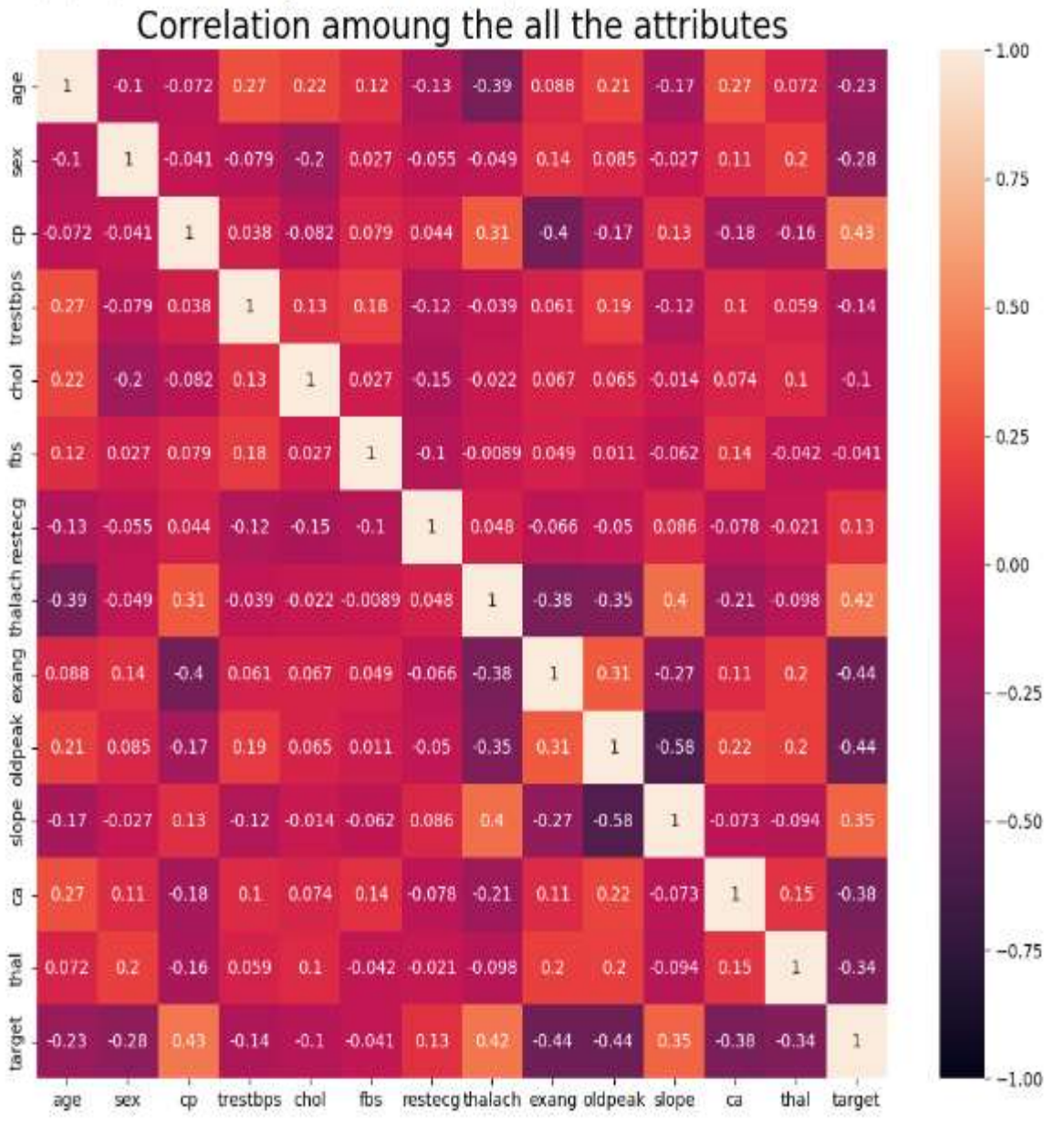


FIG 3.3 CORRELATION MATRIX

CHAPTER – 4

PROBLEM DEFINATION AND PROPOSED WORK

1. Problem Statement

The major challenge in heart disease system is it detection and prevention. There are instrument available in market which can predict heart disease but either they are very expensive or are not efficient to calculate chance of heart disease in human. Heart disease can be managed effectively with a combination of lifestyle food changes medicine and in some cases surgery is necessary because of other complication.

With the right treatment, the symptoms of heart disease can be reduced and the functional of heart can be improved. The predicted results can aid to prevent and thus reduce the cost of surgical treatment and other expensive it can be taken as early alarming stage.

The overall objective of our work will be to predict accurately with the algorithms and attributes the presence of heart disease. The decision often made based on Dr sunset and experience rather than on the knowledge which data hidden in the data set and databases.

This system will not only help doctors to protect her disease but also be patience or individual to take a third opinion about whether they have it or not.

2. Project Objectives

I. The main objectives are first to evaluate the patient is having heart disease or not on the basis of clinical records.

II. Second is to create a machine learning model that predict heart disease.

III. Third is to check efficiency on the basis of various accuracy scores.

IV. Fourth is to visualize clinical based data and reports so that its easier to analyze and work on the issue.

V. Fifth is to make work easier for both doctors and patients.

3. Proposed Work

While working on any business model or project we follow the basic steps which is shown in below diagram: -

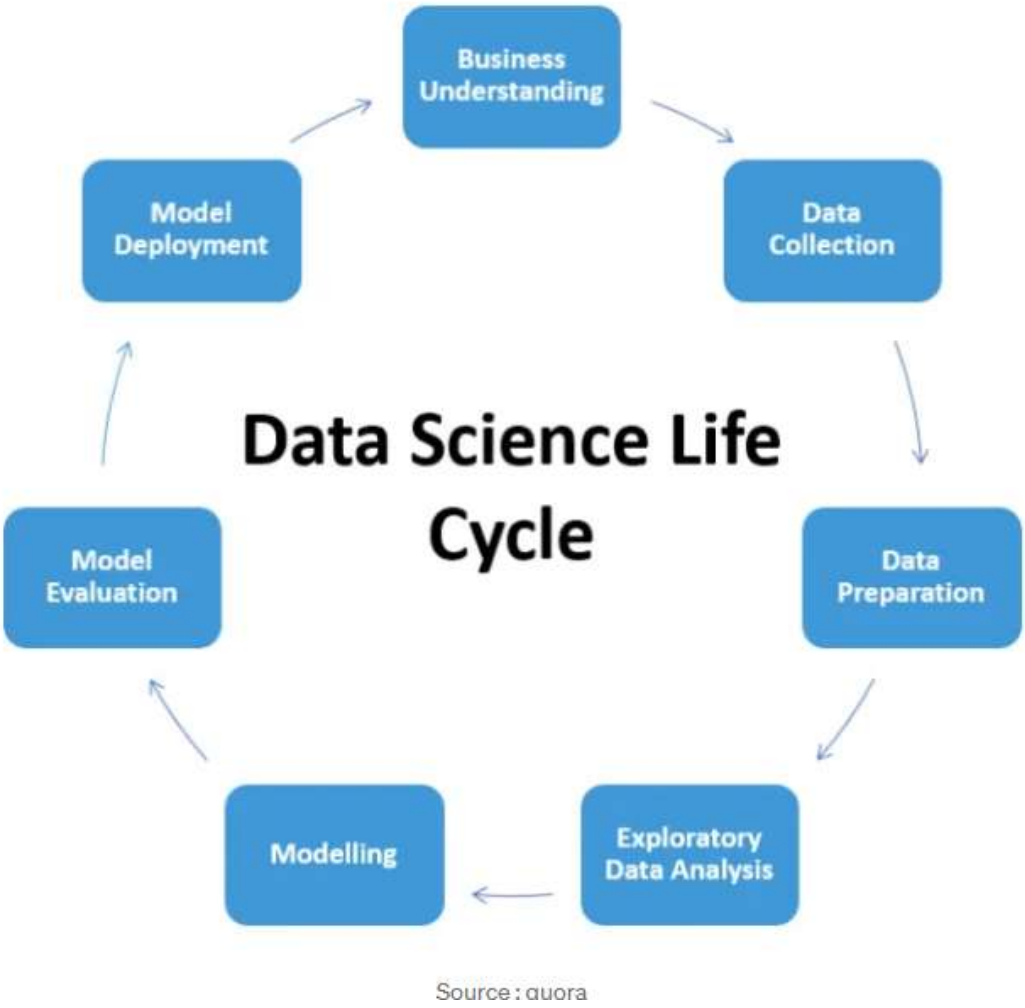


FIG 4.4 DATA FLOW

With this, the proposed work for heart disease prediction system will be

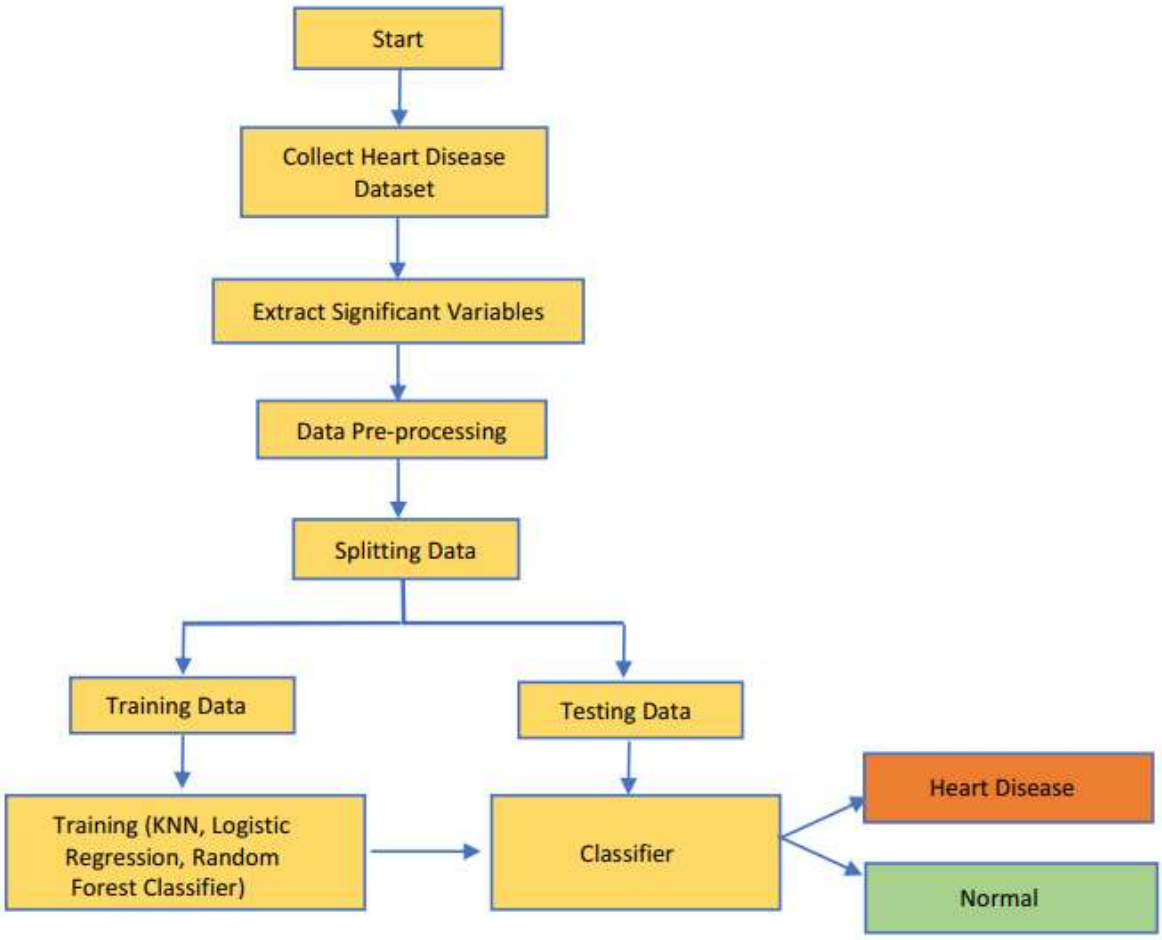


FIG 4.5 PROPOSED FLOW DIAGRAM

❖ **Retrieving Data**

The data came from Kaggle. https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset it consist of four databases.

❖ **Understanding Feature**

age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal, target.

❖ **Data pre-processing**

The data preprocessing involves of null values removing of outliers standardizing the data normalizing the data and many more.

❖ **Splitting of data**

Splitting of data refers to the practice of dividing a dataset into distinct subsets for different purposes, typically for training and evaluating machine learning models.

1. Training Set:
   - The training set is the subset of data used to train the machine learning model. The model learns patterns and relationships from this set during the training phase.

2. Test Set (or Validation Set):
   - The test set is a separate subset of data that the model has not seen during training.

Common splitting ratios include the 80-20 or 70-30 splits, where 80% or 70% of the data is used for training, and the remaining 20% or 30% is used for testing or validation.

In summary

In the realm of machine learning, the process of applying algorithms to a dataset and selecting the best-performing one involves several key steps. Initially, the dataset is divided into training and test sets to facilitate model evaluation on unseen data. Various algorithms are then applied to the training set, and their performance is assessed through metrics like accuracy, precision, recall, and F1 score. This evaluation aids in identifying the algorithm that demonstrates optimal predictive capabilities for the given task. Once the best algorithm is determined, it is applied to the entire training dataset to train the model comprehensively. This trained model is subsequently employed to make predictions on unseen data, allowing for the assessment of its generalization abilities. Rigorous testing and validation ensure that the chosen algorithm not only performs well on the training set but also exhibits robust predictive power when faced with new, previously unseen data. This iterative process of algorithm selection and model training is fundamental to creating effective and reliable machine learning models.

4. Python and python Libraries

Python is a versatile programming language that has gained immense popularity in the field of data science, machine learning, web development, and more. Several powerful libraries and frameworks contribute to Python's success in various domains. Here's a brief note on some notable Python libraries:

1. NumPy:
   - Description: NumPy is a fundamental library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with mathematical functions to operate on these arrays.
   - Use Case: It is extensively used for tasks involving numerical operations and array manipulation, forming the backbone of many scientific computing and data analysis workflows.

2. Pandas:
   - Description: Pandas is a powerful data manipulation and analysis library. It introduces data structures like DataFrame for efficient handling and manipulation of structured data.
   - Use Case: Pandas is commonly used for tasks such as data cleaning, exploration, and transformation. It's particularly helpful when working with tabular or structured data, making it an essential tool in data analysis.

3. Matplotlib and Seaborn:
   - Description: Matplotlib is a comprehensive plotting library, while Seaborn is built on top of Matplotlib and provides a high-level interface for statistical graphics.
   - Use Case: Matplotlib is widely used for creating static and interactive visualizations, while Seaborn simplifies the process of generating informative statistical graphics, making data visualization in Python more accessible.

4. Scikit-learn:
   - Description: Scikit-learn is a machine learning library that provides simple and efficient tools for data analysis and modeling, including tools for classification, regression, clustering, and more.
   - Use Case: It is extensively used for building and evaluating machine learning models. From preprocessing data to training models and evaluating their performance, Scikit-learn covers a broad spectrum of machine learning tasks.

5. Joblib:

- Description: Joblib is a library for lightweight pipelining in Python. It provides tools for parallel computing and efficient handling of tasks that involve computational-intensive operations.
- Use Case: Joblib is often used in conjunction with Scikit-learn for parallelizing tasks like cross-validation or model fitting, enhancing the computational efficiency of machine learning workflows.

6. Tkinter:

- Description: Tkinter is the standard GUI (Graphical User Interface) toolkit that comes with Python. It provides a set of tools to create desktop applications with graphical interfaces.
- Use Case: Tkinter is widely used for creating simple graphical interfaces and applications in Python. It allows developers to build interactive user interfaces for various purposes, including data visualization.

These libraries collectively form a powerful ecosystem in Python, enabling tasks ranging from data manipulation and analysis to machine learning and visualization. Their versatility and integration make them invaluable for professionals working in data science, machine learning, and related fields.

5. Google Colaboratory (Colab):

"Google Colab" or "Google Colaboratory," it is a cloud-based platform provided by Google for collaborative coding in Python. Google Colab allows users to write and execute Python code in a Jupyter Notebook environment directly in the cloud. It offers features such as free GPU access, integration with Google Drive, and the ability to share and collaborate on notebooks in real-time. Users can leverage Colab for various purposes, including data analysis, machine learning, and educational activities.

Google Colab is a cloud-based platform designed for collaborative Python programming. Key features include:

- Free GPU Access: Colab provides free access to Graphics Processing Units (GPUs) for accelerated computations, particularly useful for machine learning tasks.

- Jupyter Notebooks Integration: Users can create and share documents containing live code, visualizations, and text, fostering collaborative work.

- Pre-installed Libraries: Comes with popular Python libraries (e.g., NumPy, Pandas, TensorFlow), eliminating the need for manual installations.

- Integration with Google Drive: Seamless interaction with Google Drive for saving, sharing, and accessing Colab notebooks.

- Real-time Collaboration: Multiple users can collaborate simultaneously on the same document, facilitating teamwork.

- Cloud-Based Environment: Accessible from any device with an internet connection, with no local installations required.

CHAPTER – 5

RESULT ANALYSIS

1. **Accuracy and confusion matrix**

Accuracy:

Accuracy is a commonly used metric to evaluate the overall performance of a classification model. It is calculated as the ratio of correctly predicted instances to the total number of instances in the dataset. The formula for accuracy is:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

While accuracy is a straightforward and intuitive metric, it may not be sufficient in situations where the dataset is imbalanced (i.e., when one class significantly outnumbers the other). In such cases, other metrics like precision, recall, and the F1 score may provide a more comprehensive evaluation of the model's performance.

Confusion Matrix:

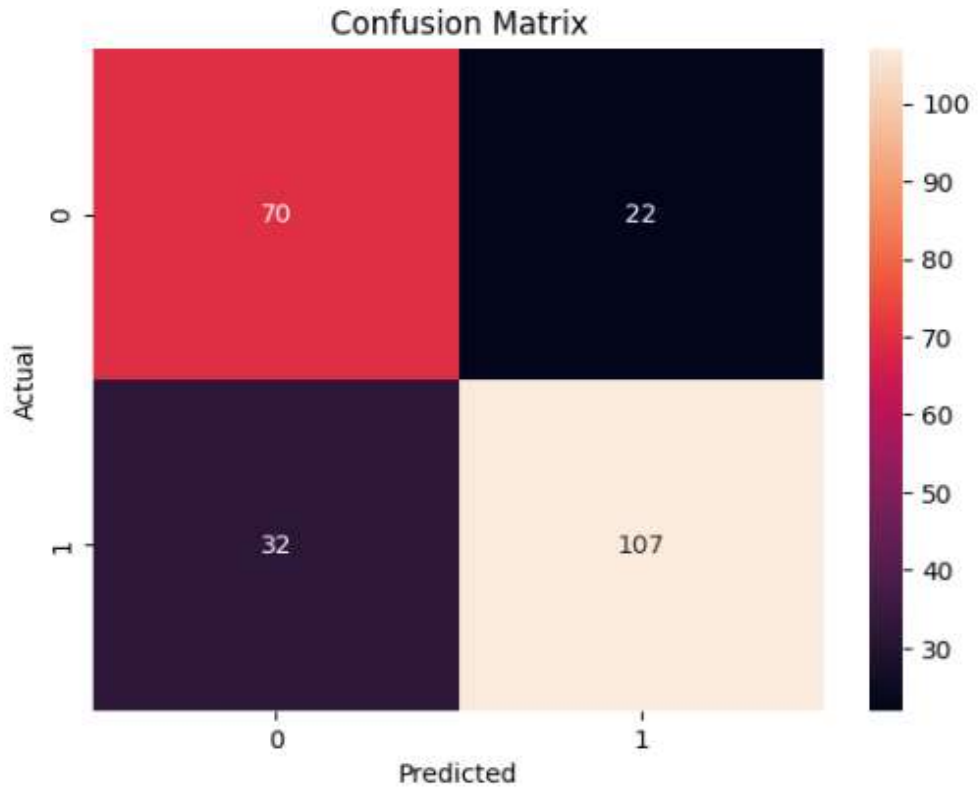
A confusion matrix is a table that is used to evaluate the performance of a classification algorithm. It provides a detailed breakdown of the model's predictions, showing the number of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). These elements are defined as follows:

- True Positives (TP): Instances where the model correctly predicts the positive class.
- True Negatives (TN): Instances where the model correctly predicts the negative class.
- False Positives (FP): Instances where the model incorrectly predicts the positive class (false alarm).
- False Negatives (FN): Instances where the model incorrectly predicts the negative class (miss).

The model we have used in this research are: -

- Naïve Bayes Algorithm:

```
Accuracy Score: 0.7662337662337663
Confusion Matrix:
[[ 70  22]
 [ 32 107]]
```

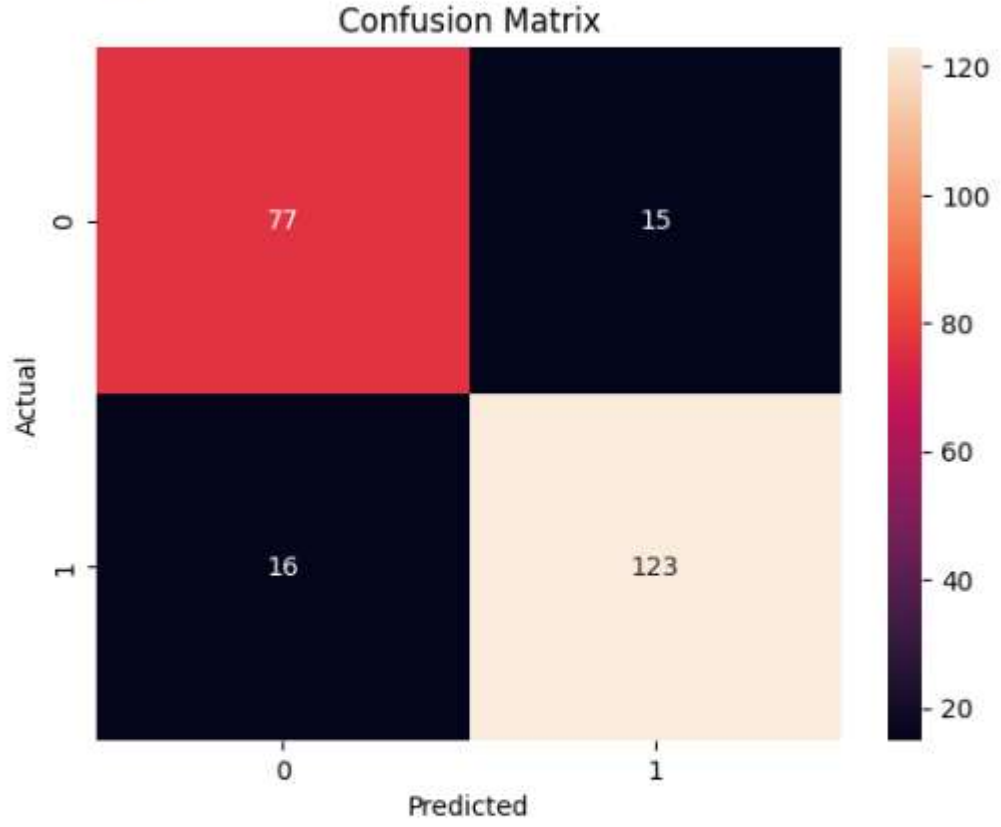



```
Classification Report:
              precision    recall  f1-score   support

         0.0       0.69      0.76      0.72        92
         1.0       0.83      0.77      0.80       139

    accuracy                           0.77       231
   macro avg       0.76      0.77      0.76       231
weighted avg       0.77      0.77      0.77       231
```

FIG 5.1 NAÏVE BAYES ALGORITHM

- Logistic Regression:

```
Accuracy Score: 0.8658008658008658
Confusion Matrix:
[[ 77  15]
 [ 16 123]]
```

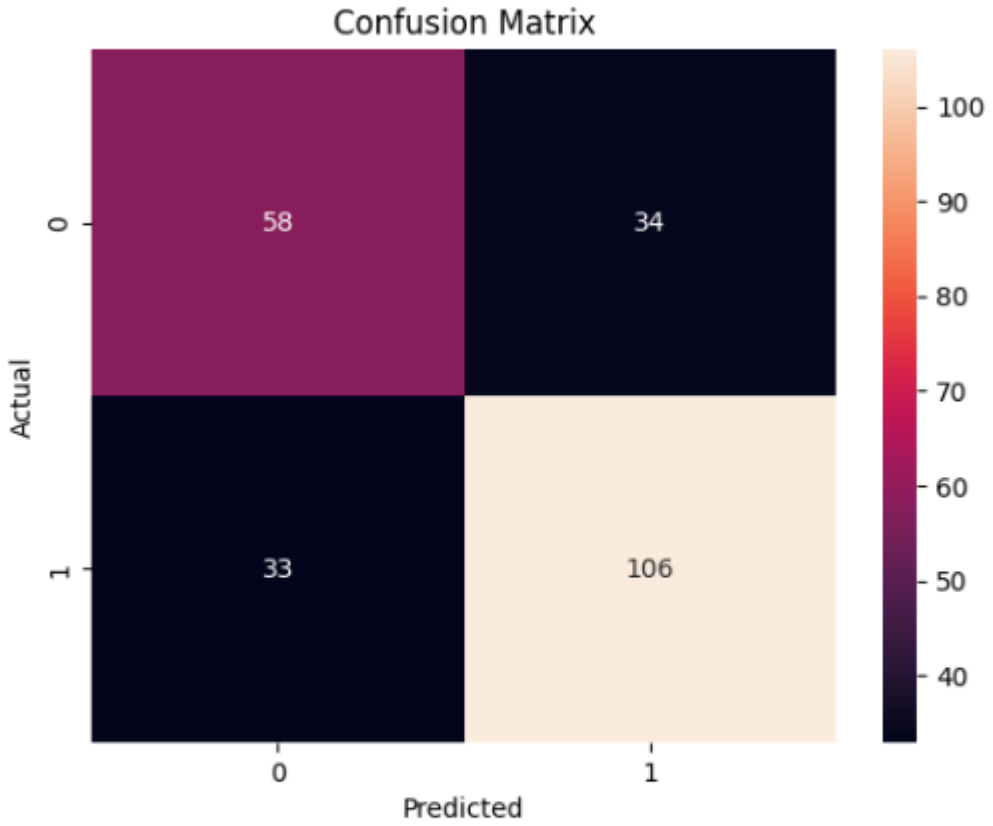



```
Classification Report:
              precision    recall  f1-score   support

         0.0       0.83      0.84      0.83        92
         1.0       0.89      0.88      0.89       139

    accuracy                           0.87       231
   macro avg       0.86      0.86      0.86       231
weighted avg       0.87      0.87      0.87       231
```

FIG 5.2 LOGISTIC REGRESSION

- K-Nearest Neighbour (KNN)

```
Accuracy Score: 0.70995670995671
Confusion Matrix:
[[ 58  34]
 [ 33 106]]
```

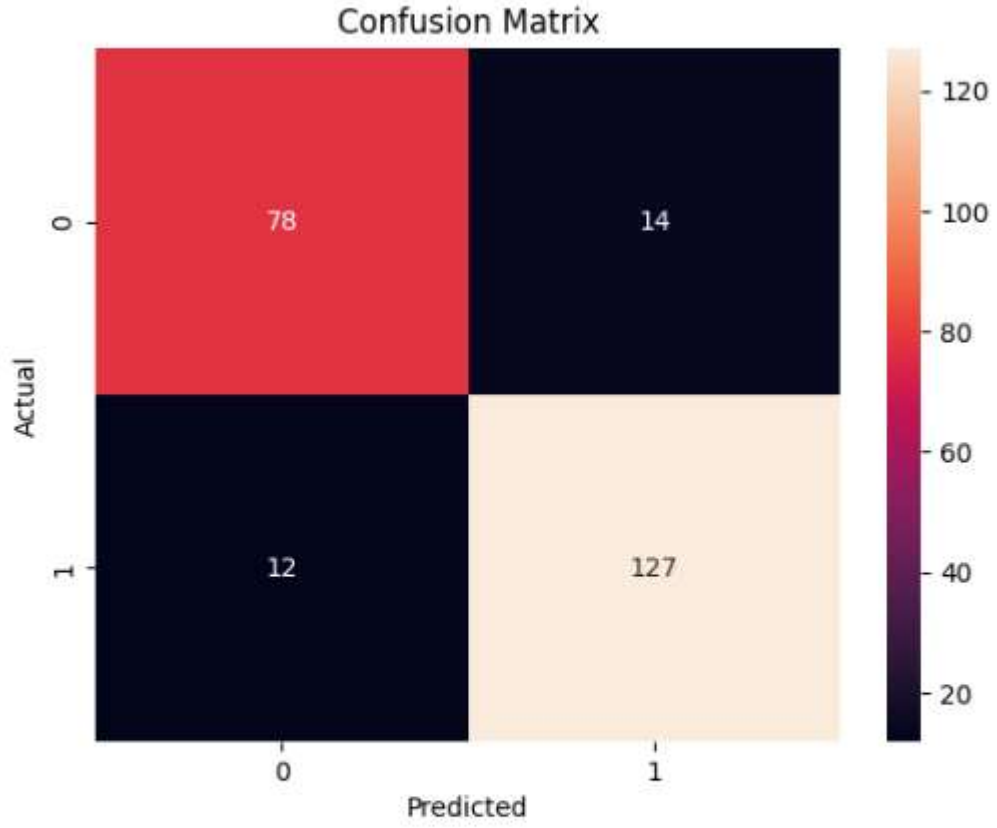



```
Classification Report:
              precision    recall  f1-score   support

         0.0       0.64      0.63      0.63        92
         1.0       0.76      0.76      0.76       139

    accuracy                           0.71       231
   macro avg       0.70      0.70      0.70       231
weighted avg       0.71      0.71      0.71       231
```

FIG 5.3 K- NEAREST NEIGHBOUR

- Decision Tree Algorithm

```
Accuracy Score: 0.8874458874458875
Confusion Matrix:
[[ 78  14]
 [ 12 127]]
```

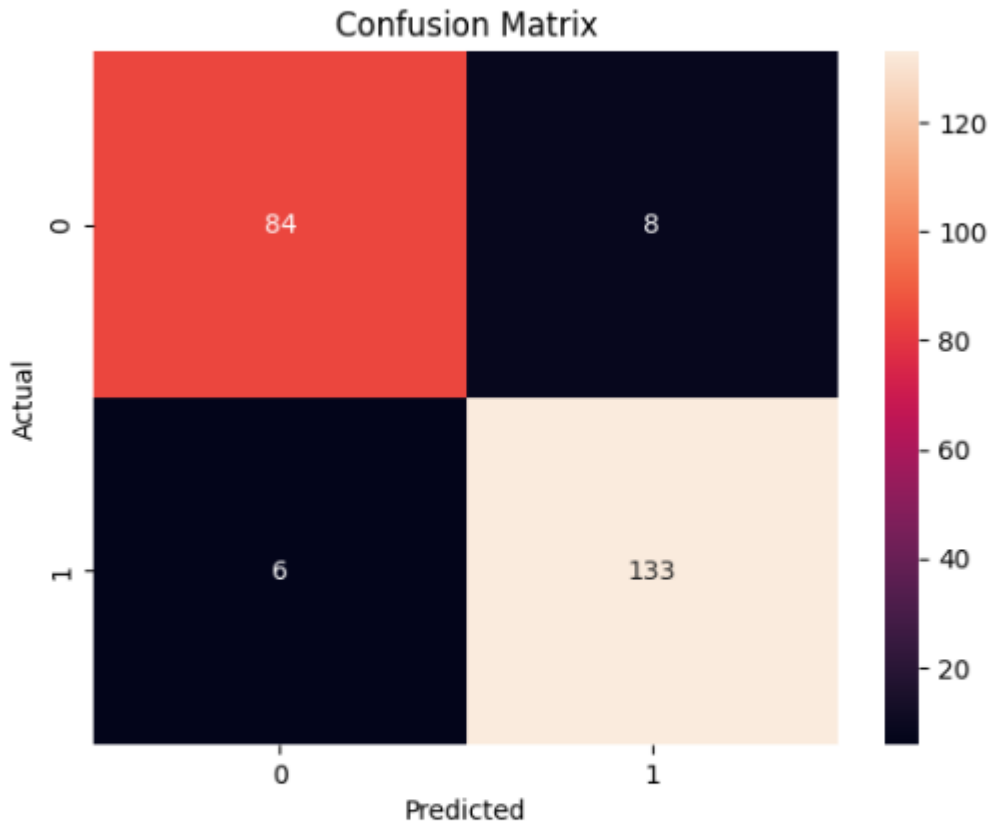



```
Classification Report:
              precision    recall  f1-score   support

         0.0       0.87      0.85      0.86        92
         1.0       0.90      0.91      0.91       139

    accuracy                           0.89       231
   macro avg       0.88      0.88      0.88       231
weighted avg       0.89      0.89      0.89       231
```

FIG 5.4 DECISION TREE

- Random Forest Algorithm

```
Accuracy Score: 0.9393939393939394
Confusion Matrix:
[[ 84   8]
 [  6 133]]
```

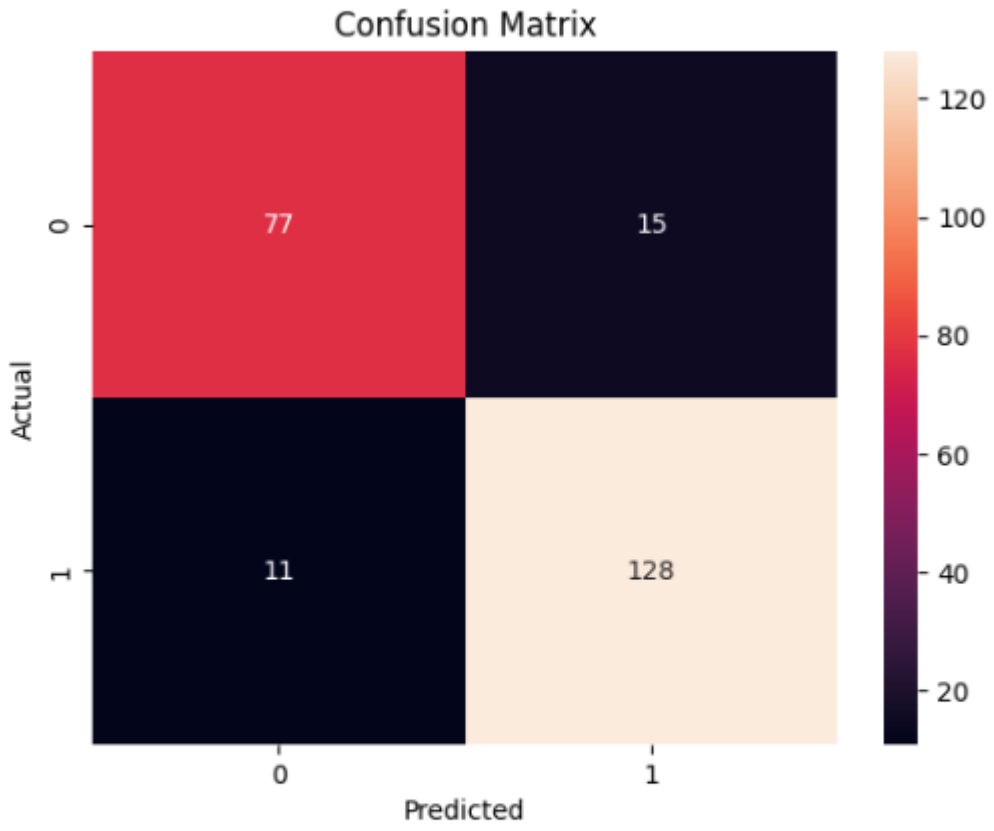



```
Classification Report:
              precision    recall  f1-score   support

         0.0       0.93      0.91      0.92        92
         1.0       0.94      0.96      0.95       139

    accuracy                           0.94       231
   macro avg       0.94      0.93      0.94       231
weighted avg       0.94      0.94      0.94       231
```

FIG 5.5 RANDOM FOREST

- Support Vector Machine Algorithm

```
Accuracy Score: 0.8874458874458875
Confusion Matrix:
[[ 77  15]
 [ 11 128]]
```




```
Classification Report:
              precision    recall  f1-score   support

         0.0       0.88      0.84      0.86        92
         1.0       0.90      0.92      0.91       139

    accuracy                           0.89       231
   macro avg       0.89      0.88      0.88       231
weighted avg       0.89      0.89      0.89       231
```

FIG 5.6 SUPPORT VECTOR MACHINE

2. Model evaluation

We can clearly see from fig 9 that random forest algorithm has the highest accuracy as compared to others.

	Models	Accuracy
0	naive bayes	76.623377
1	knn	70.995671
2	logistic regression	86.580087
3	decision tree	88.744589
4	random forest	93.939394
5	svm	88.744589

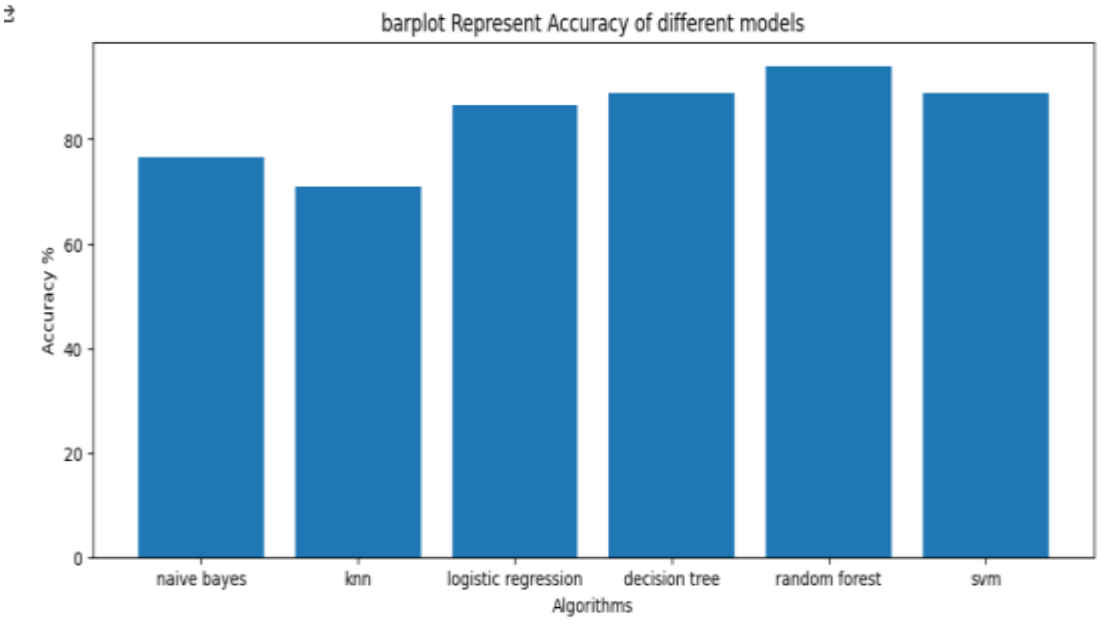


FIG 5.7 MODEL EVALUATION

CHAPTER – 6

IMPLEMENTATION

In this report, we present the development and implementation of a predictive model using Random Forest, a powerful machine learning algorithm known for its high accuracy in various applications. The model has been integrated into a Graphical User Interface (GUI) using Joblib and Tkinter to facilitate user-friendly interactions.

1. Graphical User Interface:
   A graphical user interface (GUI) is a digital interface in which a user interacts with graphical components such as icons, buttons, and menus. In a GUI, the visuals displayed in the user interface convey information relevant to the user, as well as actions that they can take. To enhance the accessibility and usability of the Random Forest model, a Graphical User Interface (GUI) was created using the Joblib library for model persistence and Tkinter for the GUI components. Tkinter is a popular Python library for creating GUI applications.

   GUI Development:

   - Tkinter Framework: Tkinter, a widely used GUI framework in Python, was employed to create the interface. It provides a set of tools for building interactive and visually appealing applications.

   - User Input Interface: The GUI allows users to input the necessary data for prediction through user-friendly input fields and widgets.

   - Model Integration: Joblib, a library for lightweight pipelining in Python, facilitated the integration of the pre-trained Random Forest model into the GUI.

   - This ensures efficient loading and execution of the model within the application.

   - Prediction Output: Once the user inputs are provided, the model processes the data and displays the prediction on the GUI interface.

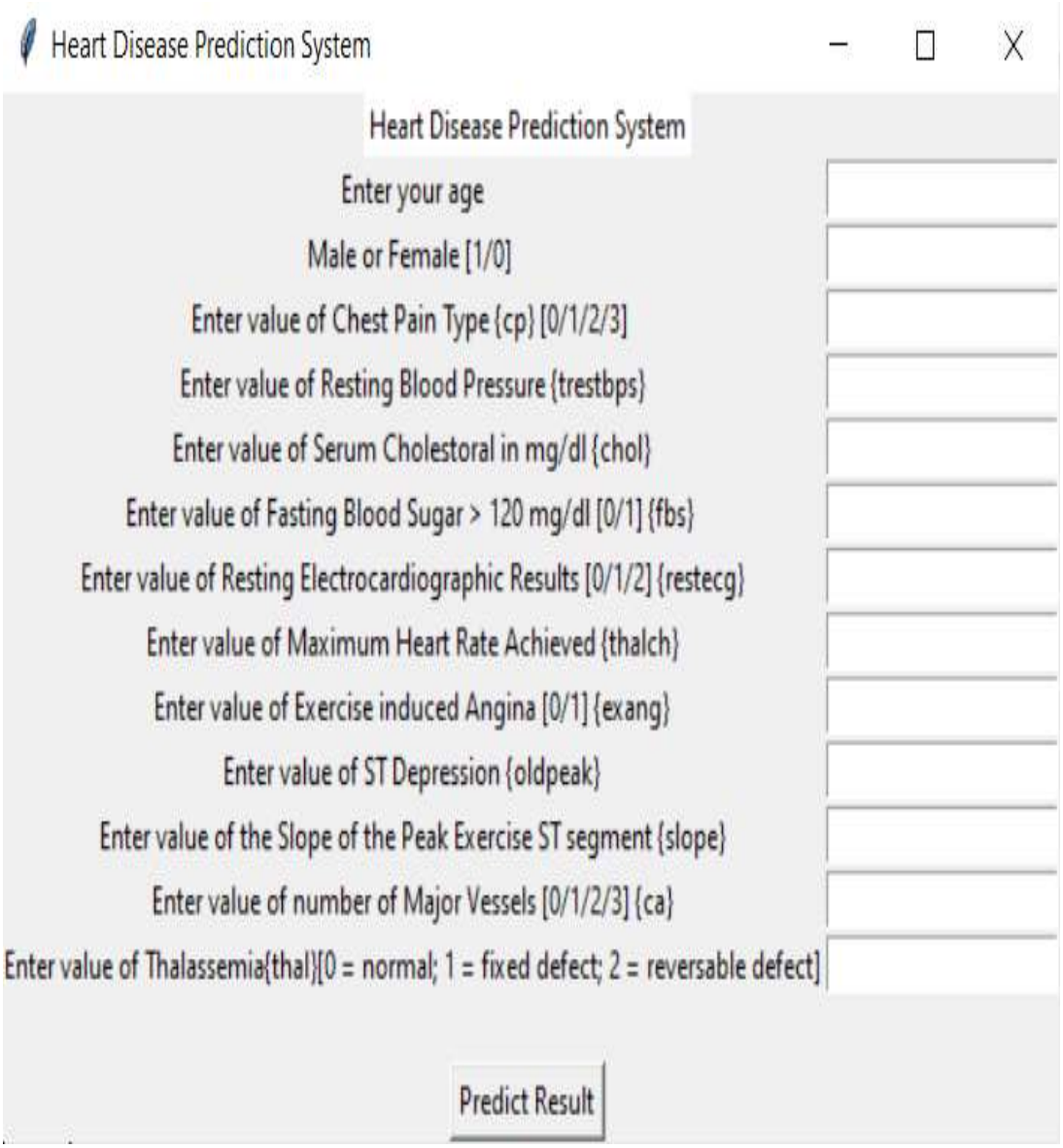


FIG 6.1 IMPLEMENTATION PART-1

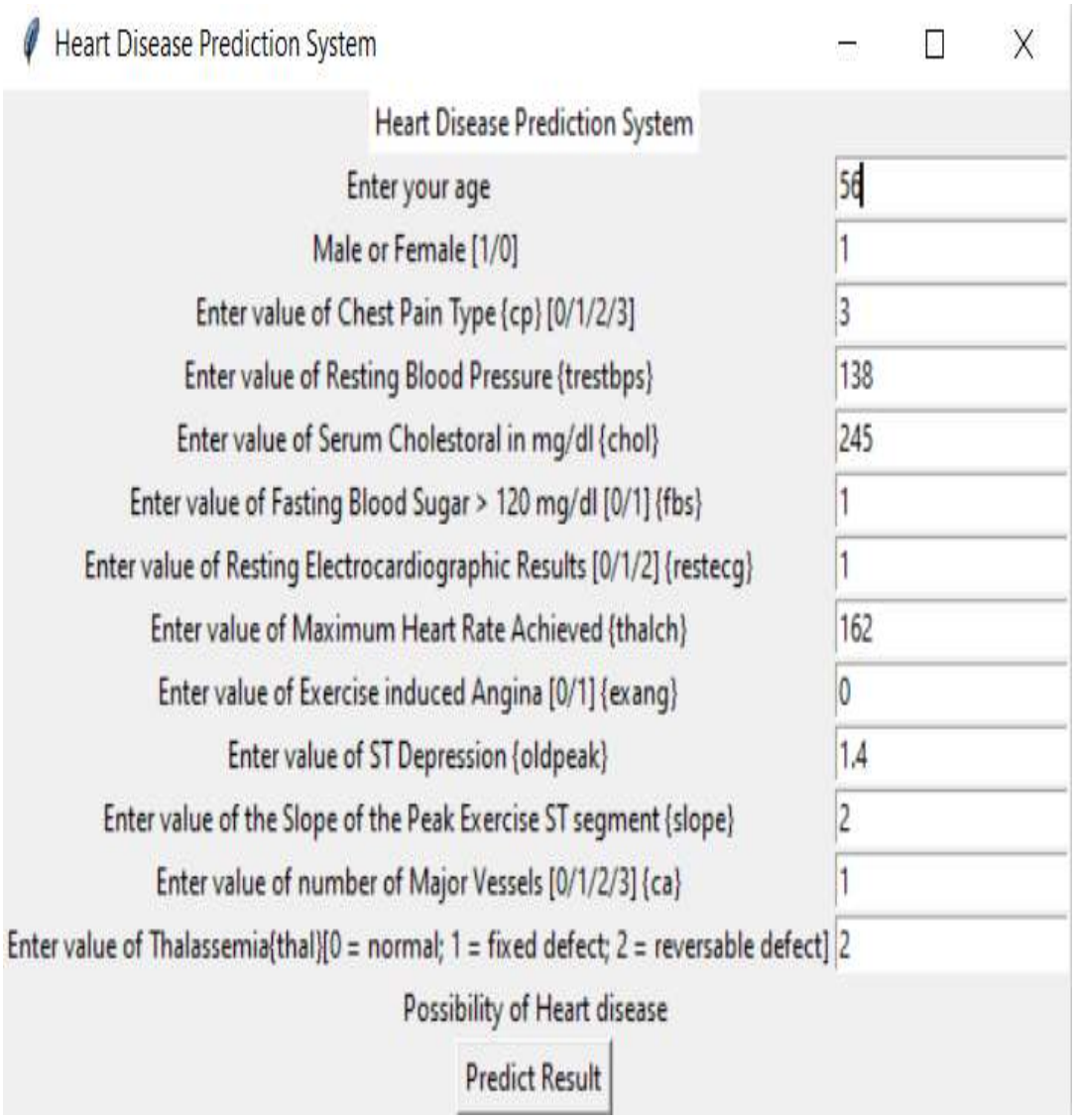


FIG 6.2 IMPLEMENTATION PART-2

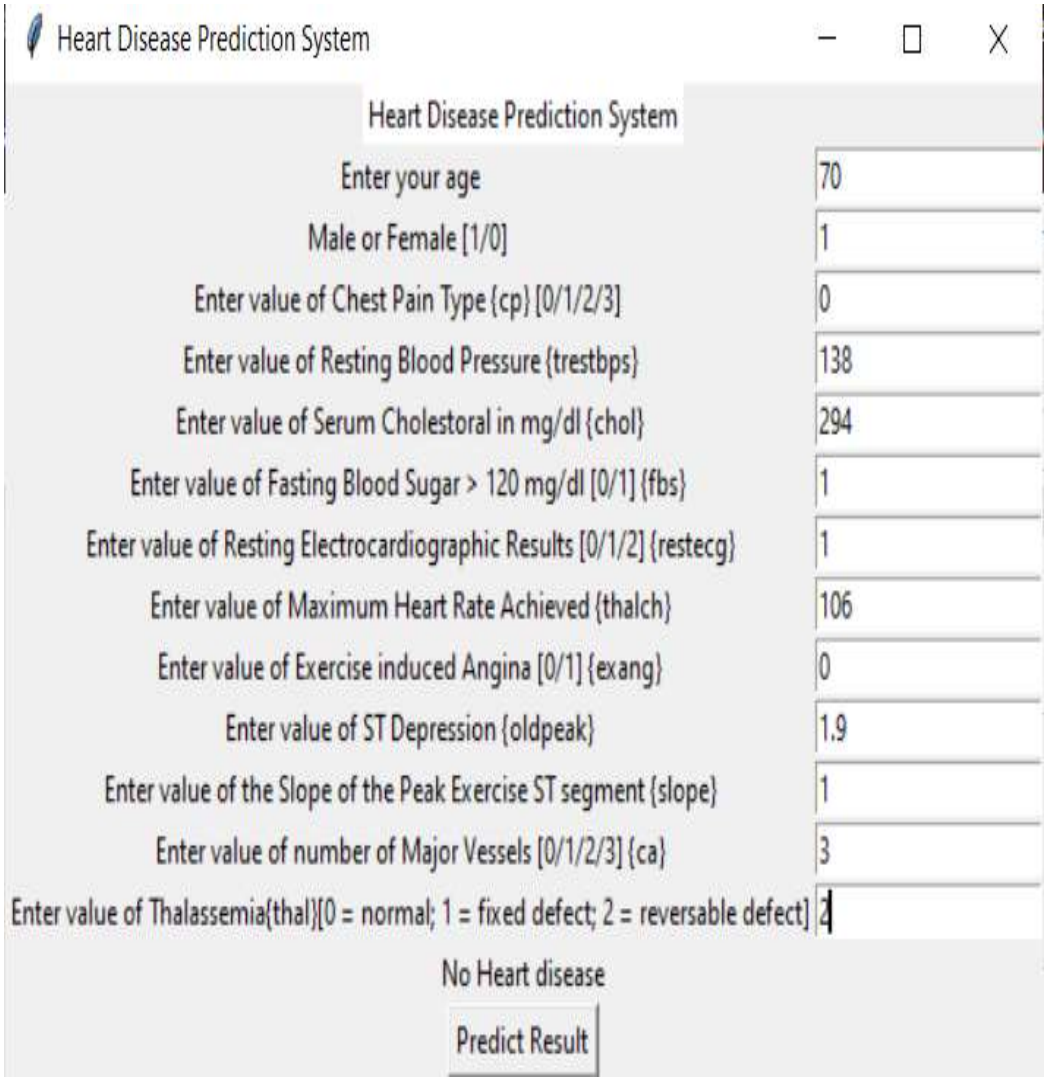


FIG 6.3 IMPLEMENTATION PART-3

CHAPTER – 7

CONCULSION

1. Conclusion

The early detection of heart disease can help in making decisions on lifestyle changes in high risks patients, internet will reduce the complication and death rate, which can be a great milestone in the field of medicine. This project predicts people with cardiovascular disease by extracting the patient's medical information that leads to a life-threatening heart disease from my data set that include patients medical history including chest pain sugar level blood pressure etc. The algorithms used in building the given model are logistic regression random forest classifier SVM decision tree and KNN. The highest accuracy of our model is 94%. Therefore, in conclusion this project helps us predict the patients who are diagnosed with heart disease by preprocessing the data set and applying various algorithm to get an accuracy of 94% on our model which is far better than the previous models having an accuracy of 85%. Also, it is concluded that accuracy of logistic regression and SVM is highest between the five algorithms that we have used that is 88% and 86%.

2. Future Work

As future research we aim to perform the following tasks:

1. We will try to increase efficiency and other evaluation matrices so that every individual can rely on the prediction work.

2. We will apply algorithms on huge data set which is having accurate data with more rows and columns which would add to increase efficiency and our model would predict heart disease more accurately.

3. Doctors and health association can be consulted to make this model a sophisticated tool for better results.

4. Ensemble learning would train model with better split and predict more true positives.

5. It would help citizens with low economic stability to analyse their clinical data with doctor's certified model.

CHAPTER – 8

REFERENCES

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

2. https://www.javatpoint.com/machine-learning-algorithms

3. https://blog.hubspot.com/website/what-is-gui#:~:text=A%20graphical%20user%20interface%20(GUI,actions%20that%20they%20can%20take.

4. https://towardsdatascience.com/what-is-feature-engineering-importance-tools-and-techniques-for-machine-learning-2080b0269f10

5. https://youtu.be/i3RMlrx4ol4?feature=shared

6. https://youtu.be/JJrnter1Sss?feature=shared

7. https://youtu.be/4LX64MDPikc?feature=shared