HEART DISEASE DETECTION

Project report submitted to

Guru Jambheshwar University of Science and Technology, Hisar

for the partial award of the degree

of

Bachelor of Technology

in

Computer Science and Engineering

by

Nishant (200010130080) **Dr. Jyoti, Professor**

Pankaj Gupta (200010130087) **Dept. of CSE**

Department of Computer Science & Engineering

GURU JAMBHESHWAR UNIVERSITY OF SCIENCE AND TECHNOLOGY, HISAR

June, 2024

CANDIDATE’S DECLARATION

We, Nishant (200010130080) & Pankaj Gupta (200010130087), certify that the work contained in this project report is original and was performed by us under the direction of our supervisor. This work has not been submitted to any other institution for the awarding of a degree, and we followed the ethical practices and other guidelines of the Department of Computer Science and Engineering in preparing our report. Whenever we use material (data, theoretical analysis, figures, text) from other sources, we cite them in the body of the report and give details in the references.

Signature

Nishant
Roll No.- 200010130080
Pankaj Gupta
Roll No.- 200010130087
Department of Computer Science and Engineering
Guru Jambheshwar University of Science and Technology, Hisar

ACKNOWLEDGEMENT

We would like to express our special thanks to our teacher who helped us to complete this project and also helped us with a lot of research, so We learned many new things. Secondly, we would also like to thank my parents, friends, and classmates who helped us a lot to complete this project within the limited time frame. Last but not least, we would like to thank everyone who directly or indirectly supported me in completing this project

CERTIFICATE

This is to certify that Nishant (200010130080) and Pankaj Gupta (200010130087) are students of B.Tech. (CSE), Department of Computer Science & Engineering, Guru Jambheshwar University of Science & Technology, Hisar has completed the project entitled "Heart Disease Detection".

Dr. Jyoti, Professor
Department of CSE
GJUS&T, Hisar

LIST OF FIGURES

CONTENTS

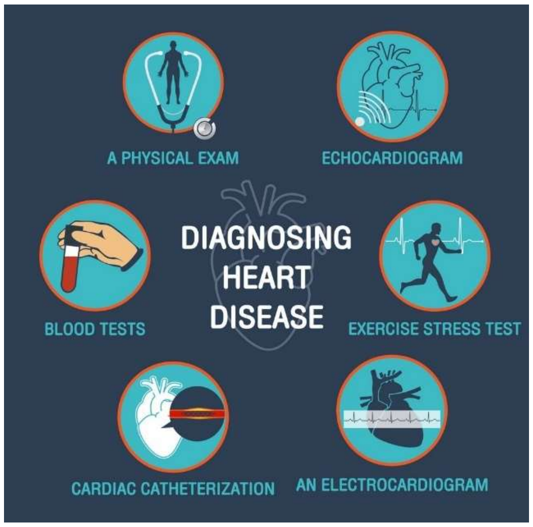
CHAPTER 1
INTRODUCTION

1.1 Introduction to Project

- Cardiovascular diseases (CVDs) are the primary cause of death globally, claiming 17.9 million lives annually on average. The early diagnosis and prevention of cardiac disorders continue to be extremely difficult tasks, notwithstanding advances in medical research. Many people don't receive a diagnosis until they suffer serious health events like heart attacks or strokes, which can have fatal results. This emphasizes how critical it is to develop efficient instruments that can quickly detect people who are at risk of heart disease, allowing for earlier interventions and improved management.
- Our goal in this research is to use cutting-edge machine-learning techniques to produce a detector for cardiac disease. Our objective is to develop a prediction model that can precisely determine the risk of heart disease by evaluating a complete collection of patient data, including physiological indicators, medical history, and demographic data. The model will help healthcare providers identify high-risk patients and suggest the best preventative interventions by acting as a decision-support tool.
- This project is significant because it has the potential to revolutionize preventative healthcare. Most importantly, early detection of cardiac disease can save lives by lowering healthcare costs and enabling more effective treatment. Our strategy leverages the most recent developments in machine learning to tackle one of the most important health concerns of our time by fusing the strength of data science with medical knowledge. The approach, application, and validation of our heart disease detector are described in this document. We will go over the procedures used to gather and prepare the data, pick pertinent features, select appropriate machine learning algorithms, and assess the model's effectiveness. We will also talk about the possible implications of our findings and ethical issues, highlighting the significance of justice and openness in predictive healthcare applications.

- Our research also bridges the gap between technology and medicine by highlighting the interdisciplinary character of contemporary healthcare solutions. Our objective is to develop a tool that is useful in clinical settings and has the potential to be both scientifically and practically sound by working with healthcare experts and using real-world data.
- Furthermore, this study emphasizes how important innovation is to public health. We hope to contribute to a future where cardiac disease can be detected and treated proactively by utilizing machine learning. By taking a proactive stance, the emphasis may be shifted from reactive therapy to preventive care, which would ultimately improve the lives of a great number of people. As we progress with this research, we're still dedicated to upholding moral principles and ensuring that our predictive model is an effective and fair tool in the battle against heart disease.
- The use of machine learning in healthcare is evidence of how patient care and medical diagnostics are developing. We can find patterns and insights that traditional methods might miss by using large datasets and complex algorithms. The goal of this study is to use these technical developments to develop a model that can accurately forecast cardiac disease and give healthcare professionals useful information.
- In the end, this research is a big step towards personalized medicine, which allows for the customization of preventive and therapeutic interventions based on the risk profiles of specific patients. Early identification of high-risk patients can facilitate prompt therapies aimed at halting the advancement of cardiac disease and enhancing patient outcomes. This is in line with the overarching objective of improving public health and lowering the prevalence of cardiovascular illnesses worldwide.
- By the project's conclusion, we want to have produced a reliable and understandable heart disease detector that can be used in clinical settings, giving medical professionals insightful information and advancing the ongoing fight against cardiovascular illnesses. This program not only shows how technology can be used to improve health outcomes but also shows how it can benefit society in a meaningful way. Our dedication to creativity, moral principles, and

teamwork will direct us as we work towards these goals and have a real influence on healthcare.



1.2 Motivation

The motivation behind developing a heart disease detector stems from the profound and growing impact of cardiovascular diseases (CVDs) on global health. Cardiovascular diseases are the leading cause of death worldwide, responsible for an estimated 17.9 million deaths each year. Despite advancements in medical science, the early detection and prevention of heart diseases remain critical challenges. Many individuals are unaware of their risk until they experience severe health events such as heart attacks or strokes, which can have devastating and often irreversible consequences. This highlights the urgent need for tools to identify at-risk individuals before significant health events occur, enabling timely and effective interventions. The following are some main reasons to put in place a traffic signal management system based on congestion:

1. **Global Health Impact:**

Cardiovascular diseases are the leading cause of death worldwide, responsible for an estimated 17.9 million deaths annually. Early detection can significantly reduce morbidity and mortality rates associated with heart disease.

2. **Proactive Healthcare:**

Early identification of high-risk individuals allows for timely interventions and preventive measures. Shifting from reactive treatment to preventive care can improve patient outcomes and quality of life.

3. **Technological Advancements**:

Machine learning algorithms can analyze complex and large datasets to uncover patterns that traditional methods might miss. Leveraging technology can enhance the accuracy and comprehensiveness of heart disease risk assessments.

4. **Economic Benefits:**

Early detection and prevention can reduce healthcare costs associated with emergency interventions and long-term treatments. Preventive measures can lower the economic burden on healthcare systems.

5. **Personalized Medicine:**

The project supports the trend towards personalized healthcare by providing tailored risk assessments and interventions based on individual profiles. Enhances the ability to deliver customized healthcare strategies for better patient management.

6. **Innovation and Research:**

Pushing the boundaries of current medical research and innovation through the application of advanced machine learning techniques. Providing a foundation for future research and development in predictive healthcare tools.

CHAPTER 2
BACKGROUND DETAILS AND LITERATURE REVIEW

2.1 Background:

Traditionally, the diagnosis and assessment of heart disease risk have relied heavily on clinical evaluation and basic diagnostic tests, such as blood pressure measurement, cholesterol levels, and electrocardiograms (ECGs). While these methods are valuable, they often fall short in detecting heart disease at an early stage or in predicting future cardiovascular events. Moreover, they may not fully account for the complex interplay of genetic, lifestyle, and environmental factors that contribute to heart disease risk.

Recent advancements in data science and machine learning have opened new avenues for improving heart disease detection and risk assessment. Machine learning algorithms have the potential to analyze large, complex datasets to identify patterns and relationships that are not apparent through traditional statistical methods. By leveraging data from electronic health records (EHRs), wearable devices, and other sources, these algorithms can provide more accurate and personalized predictions of heart disease risk.

2.2 Literature Review:

1. Traditional Risk Assessment Models:

Early approaches to heart disease prediction relied heavily on statistical models such as the Framingham Risk Score (FRS) and the SCORE (Systematic Coronary Risk Evaluation) system. The Framingham Risk Score, developed in the 20th century, uses a range of risk factors including age, cholesterol levels, blood pressure, smoking status, and diabetes to estimate the 10-year cardiovascular risk of an individual. Similarly, the SCORE system, used widely in Europe, estimates the 10-year risk of a fatal cardiovascular event. While these models have been instrumental in clinical practice, they are limited by their reliance on a relatively small number of risk factors and their inability to account for complex interactions between variables.

2. Machine Learning in Healthcare:

The advent of machine learning (ML) has revolutionized predictive modeling in healthcare. Machine learning algorithms can analyze vast amounts of data and uncover patterns that are not apparent through traditional statistical methods. Techniques such as decision trees, random forests, support vector machines (SVM), and neural networks have been applied to predict heart disease risk with promising results.

4. Interpretability and Explainability:

One of the challenges with advanced machine learning models, particularly neural networks, is their lack of interpretability. Researchers have employed techniques such as Shapley Additive explanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME) to make model predictions more understandable. Lundberg and Lee (2017) demonstrated the application of SHAP values in healthcare, providing insights into which features most significantly impact the model's predictions.

5. Ethical and Practical Considerations:

Ethical considerations are paramount in developing predictive models for healthcare. Studies have highlighted the importance of data privacy, and security, and the need to avoid biases that could lead to health disparities. Obermeyer et al. (2019) discussed the potential for algorithmic bias in healthcare and emphasized the need for rigorous validation and fairness assessments to ensure equitable outcomes.

The literature on heart disease prediction demonstrates significant progress in leveraging machine learning to enhance traditional risk assessment models. Advanced algorithms, comprehensive data integration, and interpretability techniques have collectively contributed to more accurate and actionable predictions. However, challenges remain, particularly in ensuring ethical use and maintaining model transparency. Future research should focus on refining these models, addressing ethical concerns, and integrating them into clinical practice to maximize their potential to improve cardiovascular health outcomes.

By building on these advancements, the current project aims to develop a robust and interpretable heart disease detector that leverages state-of-the-art machine learning techniques to provide accurate and actionable risk assessments, ultimately contributing to better preventive care and management of heart disease.

2.3 Existing Systems:

Heart disease prediction systems have come a long way, from simple statistical models to sophisticated machine-learning algorithms. Here is a summary of some of the most prominent heart disease prediction systems now in use

2.3.1 Framingham Risk Score (FRS)

The Framingham Heart Study gave rise to the Framingham Risk Score, which is now one of the most popular and ancient methods for estimating the risk of cardiovascular illnesses.

Limitations:

- Limited to the factors included in the study, potentially omitting relevant variables.
- May not be as accurate in diverse populations outside the original study group.

2.3.2 Systematic Coronary Risk Evaluation (SCORE)

Developed by the European Society of Cardiology, the SCORE system predicts the 10-year risk of fatal cardiovascular disease.

- Factors considered include age, sex, smoking status, total cholesterol, and systolic blood pressure.
- Provides different charts for high-risk and low-risk European countries.

Limitations:

- May not capture non-fatal cardiovascular events.
- Potentially less applicable to non-European populations.

2.3.3 QRISK

The QRISK model is used in the UK to estimate the 10-year risk of developing cardiovascular disease. It has undergone several iterations, with QRISK3 being the latest version.

- Includes traditional risk factors along with additional factors such as ethnicity, family history, chronic kidney disease, atrial fibrillation, and socioeconomic status.
- Uses data from general practice records, making it highly relevant for primary care.

Limitations:

- Primarily developed for the UK population, which may limit its applicability elsewhere.
- Complexity in integrating and interpreting numerous risk factors.

2.3.4 Electronic Health Record (EHR)-Integrated Systems

Several health systems have integrated predictive models into their electronic health records (EHRs) to provide real-time risk assessments.

Mayo Clinic's Cardiovascular Risk Calculator: Utilises predictive modeling with EHR data.

Cleveland Clinic's Risk Assessment Tool: adapts treatment depending on each patient's unique risk profile.

Limitations:

- Dependence on the quality and completeness of EHR data.
- Challenges in standardizing data from different EHR systems.

Existing systems for heart disease prediction range from traditional statistical models like the Framingham Risk Score and SCORE to advanced machine learning-based approaches. Each system has its strengths and limitations, with newer models offering enhanced accuracy and comprehensiveness but often at the cost of interpretability and ease of use. The integration of machine learning into heart disease prediction holds significant promise for improving early detection and preventive care, but it also requires careful consideration of data quality, model transparency, and ethical implications.

CHAPTER 3
OBJECTIVE AND PROBLEM FORMULATION

3.1 Problem Statement

Heart disease remains a leading cause of morbidity and mortality worldwide, affecting millions of people and posing significant challenges to healthcare systems. Early detection and diagnosis of heart disease are crucial for effective treatment and management, potentially saving lives and reducing healthcare costs.Traditional diagnostic methods, though effective, often require invasive procedures, extensive testing, and expert interpretation. There is a growing need for non-invasive, accurate, and efficient diagnostic tools that can aid healthcare professionals in identifying heart disease at an early stage.

- To address this, we aim to develop a machine learning-based heart disease prediction model that utilizes comprehensive patient data to provide accurate and individualized risk assessments.

CHAPTER 4
SOFTWARE REQUIREMENTS SPECIFICATION

4.1 SOFTWARE REQUIREMENTS:

4.1.1 Operating System: Windows7/8/10

4.1.2 Programming Language:

- **Python:** Python is chosen for this project due to its extensive libraries for data science and machine learning, ease of use, and strong community support.

4.1.3 Jupyter Notebook

An interactive development environment that allows for the combination of code execution, rich text, mathematics, plots, and rich media into a single document. It is ideal for data analysis, visualization, and interactive development.

- **Interactive Coding:**
  Real-time code execution with immediate feedback.

- **Rich Media Support:**
  Integrates images, videos, and LaTeX equations.

- **Multi-language Support:**
  Supports over 40 programming languages, including Python, R, and Julia.

- **Data Visualization:**
  Integrates with visualization libraries like Matplotlib and Seaborn.

- **Narrative Text:**
  Uses Markdown and HTML for documentation.

- **Modular Code Organization:**
  Organizes code into cells for easier testing and debugging.

- **Easy Sharing:**
  Shareable via GitHub, email, or export to HTML, PDF, etc.

- **Extensions and Customization:**
  Supports extensions for additional functionalities.

- **Interactive Widgets:**
  Creates dynamic and interactive data exploration tools.

4.1.4 Anaconda Distribution:

- **Comprehensive Package and Environment Management:**

  Anaconda includes 'conda', a robust package manager that simplifies the installation, update, and management of dependencies. It also allows for the creation of isolated environments, ensuring compatibility and avoiding conflicts across projects.

- **Pre-installed Data Science Libraries and Tools:**

  Anaconda comes with numerous pre-installed libraries essential for data science and machine learning, such as NumPy, Pandas, Matplotlib, Seaborn, Scikit-Learn, Statsmodels, and Jupyter Notebook, enabling users to start their projects immediately without additional setup.

4.2 HARDWARE REQUIREMENTS

4.2.1 Processor: Minimum Intel i5 or equivalent.

4.2.2 Ram: 8 GB

4.2.3 SSD:128 GB

4.2.4 Hard Disk: 256 GB

4.2.5 Input device: Standard Keyboard and Mouse

4.2.6 Output device: High-Resolution Monitor.

4.2.7 Graphics Processing Unit (GPU):

High-performance GPU for machine learning tasks.

CHAPTER 5
METHODOLOGY AND TOOLS USED

5.1 Methodology

The heart disease detection project follows a structured methodology to ensure systematic development and evaluation of the machine learning model. The process involves the following key steps:

1. **Data Collection and Preprocessing:**
    - **Data Collection:** Gather a comprehensive dataset containing patient medical records with features such as age, sex, blood pressure, cholesterol levels, and heart disease diagnosis.
    - **Data Cleaning:** Handle missing values, remove duplicates, and correct any inconsistencies in the data.
    - **Data Transformation:** Standardize and normalize data to ensure consistency. Convert categorical variables into numerical values using techniques like one-hot encoding.
2. **Exploratory Data Analysis (EDA):**
    - **Statistical Analysis:** Compute summary statistics to understand the distribution of data.
    - **Visualization:** Create visualizations (e.g., histograms, box plots, correlation heatmaps) to identify patterns, trends, and correlations among features.
    - **Feature Engineering:** Derive new features or modify existing ones to improve model performance.
3. **Feature Selection:**
    - **Correlation Analysis:** Identify features that are highly correlated with the target variable (heart disease) and those that are redundant.

- **Feature Importance:** Use algorithms like Random Forest and techniques such as Recursive Feature Elimination (RFE) to select the most important features.

4. **Model Development:**

   - **Model Selection:** Implement various machine learning algorithms, including:

     - **Logistic Regression:**

       Logistic Regression is a statistical method used for binary classification tasks like predicting heart disease. It calculates the probability of an event occurring based on input variables using a logistic function. This model is valued for its simplicity, interpretability through coefficients that show feature importance, and efficiency with smaller datasets. However, it assumes a linear relationship between predictors and outcomes, which may limit its accuracy with complex data patterns and require predictors to be independent. Nonetheless, Logistic Regression remains a foundational tool in medical diagnostics and risk assessment due to its straightforward implementation and meaningful interpretative insights.

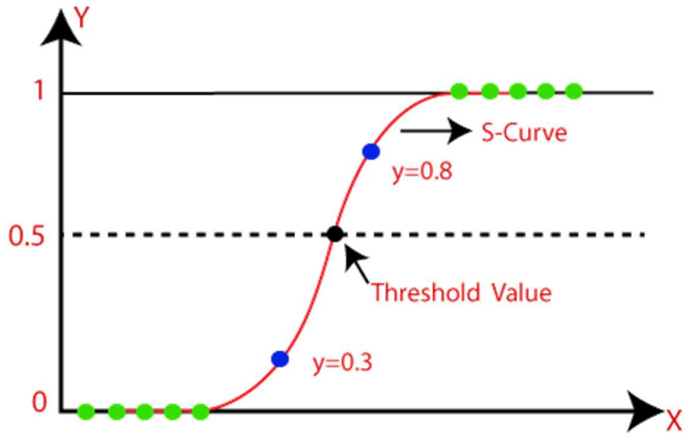


Fig 5.1: Logistic Regression Algorithm

- **Decision Trees:**

  A Decision Tree Classifier is a versatile machine learning algorithm that partitions the feature space into segments based on feature values. Each decision (split) is made at nodes, directing the data down different branches until reaching leaf nodes, which represent class labels. It excels in interpretability, as each path from root to leaf is a clear set of rules based on features. However, decision trees can overfit noisy data, so techniques like pruning or limiting tree depth are used to enhance generalization. They are foundational in machine learning for their ability to handle both numerical and categorical data, making them suitable for various tasks from healthcare diagnostics to customer segmentation in marketing.

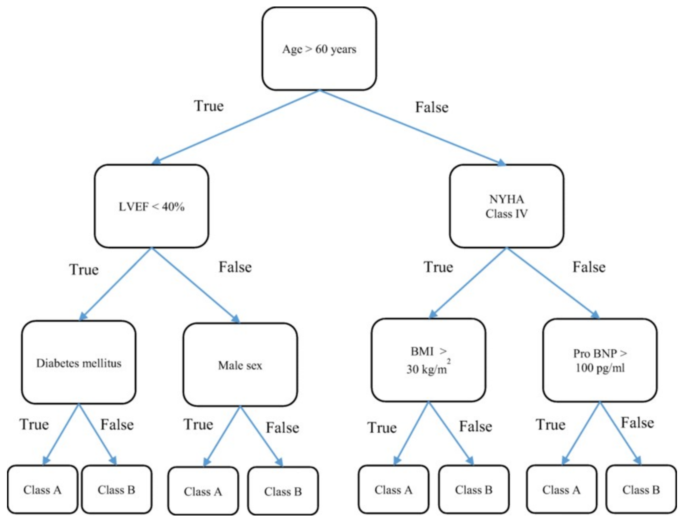


Fig 5.2: Decision Tree Classifier

- **Random Forest:**

  Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. It improves accuracy and reduces overfitting by combining the predictions of multiple trees trained on different random samples of the dataset. Random Forests are widely used due to their robust performance across various domains, ability to handle large datasets with high dimensionality, and feature importance estimation.

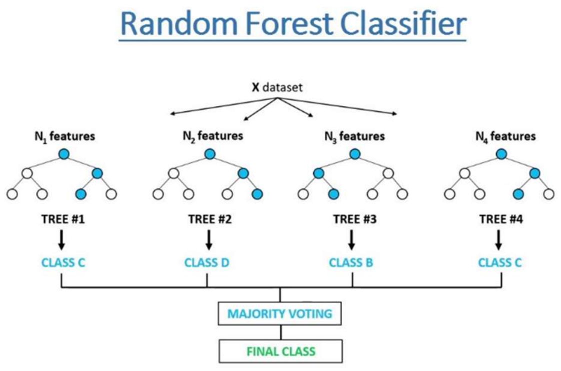


Fig 5.3: Random Forest Classifier

- **Support Vector Machines (SVM):**

  A supervised learning algorithm that finds the optimal hyperplane to separate data into classes. It is effective in high-dimensional spaces and uses kernel functions to handle non-linear relationships.

  Linear SVM:

  - Efficiently separates classes with a straight line, plane, or hyperplane in higher dimensions.
  - Suitable for large-scale datasets and high-dimensional spaces.

    Best for scenarios where classes are linearly separable.

Non-linear SVM:

- Uses kernel functions to map data into higher-dimensional spaces.
- Handles complex, non-linear relationships between features and targets.
- Enhances classification accuracy on datasets that are not linearly separable.

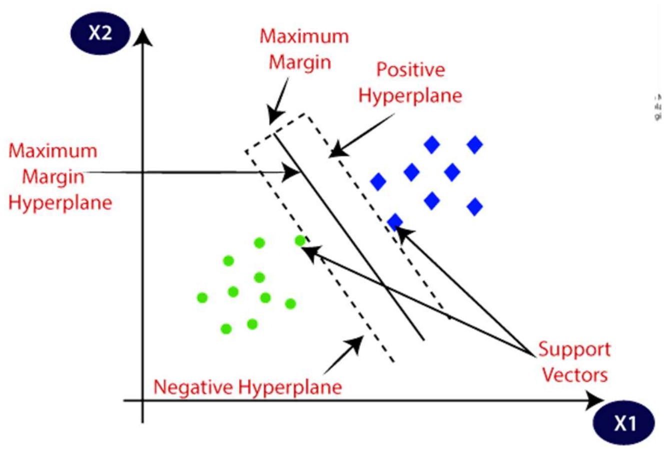


Fig 5.4: Support Vector Machines

- **Training and Validation:** Split the dataset into training and validation sets. Train models on the training set and evaluate their performance on the validation set.

5. **Model Evaluation:**

   - **Evaluation Metrics:** Assess model performance using metrics such as accuracy, precision, recall, F1 score, and ROC-AUC.
   - **Cross-Validation:** Perform k-fold cross-validation to ensure the model generalizes well to unseen data.
   - **Hyperparameter Tuning:** Optimize model hyperparameters using techniques like Grid Search or Random Search to enhance model performance.

6. **Model Interpretation:**

    - **Interpretability:** Analyze model outputs and feature importance to understand how predictions are made.
    - **Visualization:** Use techniques like SHAP (SHapley Additive exPlanations) values to interpret complex models.

7. **Model Deployment:**

    - **User Interface:** Develop an intuitive interface (e.g., web application) for healthcare professionals to input patient data and receive real-time predictions.
    - **Integration:** Ensure the model integrates seamlessly into the existing healthcare workflow.

5.2 Libraries Used:

For a heart disease detection model using machine learning in Python, several libraries are typically used to handle data manipulation, model building, evaluation, and visualization. Here are some commonly used libraries:

5.2.1 NumPy

- **Functionality**: NumPy is fundamental for numerical computing in Python, providing support for large, multi-dimensional arrays and matrices. It includes a vast library of mathematical functions to operate on these arrays efficiently.
- **Usage**: In the heart disease detection model, NumPy is used for data preprocessing tasks such as handling missing values, scaling features, and performing matrix operations required by machine learning algorithms.
- **Example**: Calculating statistical measures like mean, median, and standard deviation of features to understand data distributions and make informed preprocessing decisions.

5.2.2 Pandas

- **Functionality**: Pandas offers high-performance data structures like DataFrame, which are designed for efficient data manipulation and analysis. It provides tools for reading and writing data, handling missing values, and performing data aggregation and transformation operations.
- **Usage**: In the heart disease detection project, Pandas is used to load datasets into DataFrames, explore and clean data, perform feature engineering, and prepare data for model training.
- **Example**: Merging datasets, selecting subsets of data based on conditions, and grouping data for summary statistics to gain insights into the dataset.

5.2.3 Matplotlib

- **Functionality**: Matplotlib is a comprehensive plotting library in Python that generates static, animated, and interactive visualizations. It supports a wide variety of charts and graphs for data visualization tasks.
- **Usage**: Used extensively in the heart disease detection model to create visual representations of data distributions, relationships between variables, and model evaluation metrics.
- **Example**: Plotting histograms to visualize the distribution of features like age and cholesterol levels, scatter plots to explore correlations between variables, and ROC curves to evaluate model performance.

5.2.4 Seaborn

- **Functionality**: Seaborn is built on top of Matplotlib and provides a high-level interface for drawing attractive and informative statistical graphics. It simplifies the creation of complex visualizations with concise syntax.
- **Usage**: Utilized in the heart disease detection project to create advanced statistical plots such as heatmaps, pair plots, and categorical plots to explore relationships and patterns in the data.
- **Example**: Creating heatmaps to visualize correlations between features, pair plots to examine pairwise relationships across multiple variables, and categorical plots to compare distributions of categorical data.

5.2.5 Scikit-learn

- **Functionality**: Scikit-learn is a machine learning library in Python that provides efficient tools for data mining and data analysis. It offers a wide range of supervised and unsupervised learning algorithms, data preprocessing utilities, and model evaluation tools.
- **Usage**: Central to the heart disease detection model for implementing machine learning algorithms such as SVM, Decision Trees, and Random Forests for classification tasks. Also used for data preprocessing steps like feature scaling and dimensionality reduction.
- **Example**: Training SVM models to classify patients as having heart disease or not based on clinical data, using GridSearchCV to tune hyperparameters, and evaluating model performance with cross-validation techniques.

5.2.6. Statsmodels

- **Functionality**: Statsmodels is a Python library for estimating and testing statistical models. It includes tools for linear regression, time series analysis, and hypothesis testing.
- **Usage**: Applied in the heart disease detection project for statistical modeling tasks, such as exploring relationships between variables, conducting hypothesis tests, and fitting regression models to understand predictors of heart disease.
- **Example**: Performing logistic regression to assess the influence of risk factors like age, blood pressure, and cholesterol levels on the likelihood of heart disease, and conducting statistical tests to validate model assumptions.

5.2.7 Tkinter

- **Functionality**: Tkinter is the standard GUI library for Python, providing widgets and utilities for building desktop applications with graphical user interfaces.
- **Usage**: Integrated into the heart disease detection model to develop a user-friendly interface for inputting patient data, displaying model predictions, and visualizing results interactively.

- **Example**: Creating buttons, entry fields, and text boxes to design a GUI application where users can input medical data, trigger model predictions, and view diagnostic outcomes.

These libraries collectively form a robust ecosystem for developing, training, evaluating, and deploying machine learning models for heart disease detection. Each plays a crucial role in different stages of the data science pipeline, from data preprocessing and exploration to model building, evaluation, and visualization of results. Their integration enables efficient development and deployment of predictive models while ensuring scalability, reliability, and interpretability in healthcare applications.

CHAPTER 6
SYSTEM DESIGN

6.1 System Overview:

A general overview of Heart Disease Detector is given below:

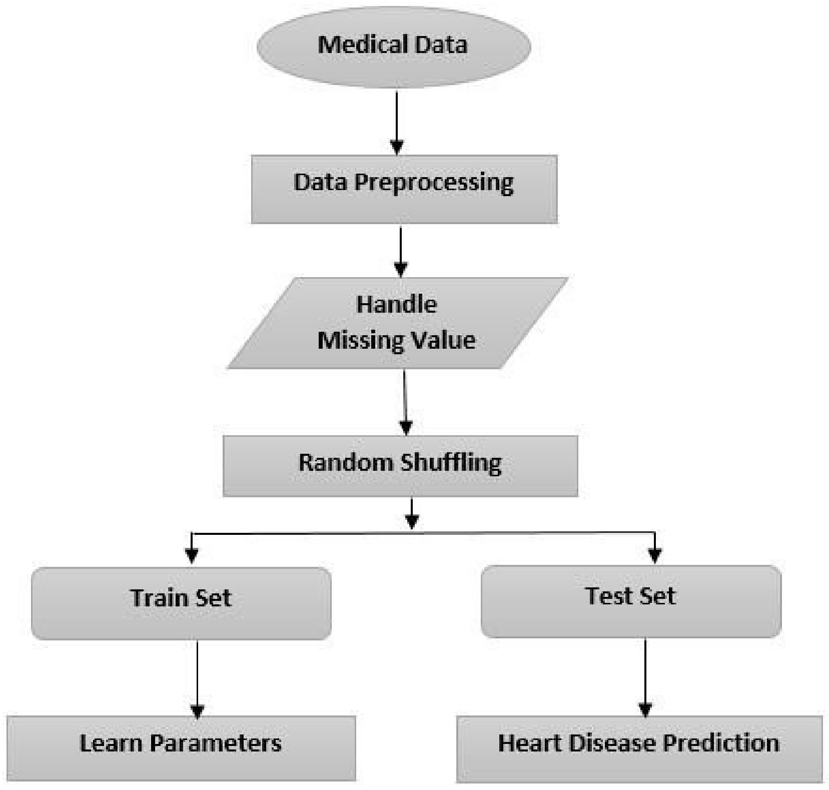


Figure 6.1: Architecture of the System

6.1.1 Medical Data

I obtained a heart disease dataset from Kaggle, which includes anonymized patient information such as age, sex, blood pressure measurements, cholesterol levels, and various clinical symptoms. The dataset aims to predict the presence of heart disease based on these features, making it suitable for developing machine-learning models for diagnostic purposes.

6.1.2 Data Preprocessing

- **Data Cleaning**:
  - **Identifying Outliers**: Detect and handle outliers that may skew statistical analyses or model predictions. Techniques include using statistical methods like Z-score or IQR (Interquartile Range).
  - **Ensuring Consistency**: Standardize formats and units across datasets to maintain data integrity and facilitate accurate analysis.
- **Null Value Removal**:
  - **Identifying Null Values**: Identify fields with missing data, typically represented as NaN (Not a Number) or Null values.
  - **Strategies for Removal**: Remove rows or columns with significant null values if they cannot be imputed or replaced with meaningful values.
  - **Imputation Techniques**: Employ methods such as mean, median, mode imputation, or more sophisticated techniques like predictive modeling (using algorithms like K-Nearest Neighbors or decision trees) to fill missing values.

6.1.3 Handle Missing Value

Handling missing values is crucial in data preprocessing to ensure data quality and integrity for effective analysis and modeling:

1. **Identification**: Identify fields with missing values, typically denoted as NaN or Null.
2. **Strategies**:
   - **Imputation**: Fill missing values using statistical measures like mean, median, or mode for numerical data, or using forward/backward filling for time series data.
   - **Deletion**: Remove rows or columns with significant missing data if imputation is not feasible or may introduce bias.
3. **Impact**: Choose methods based on data distribution and the importance of missing values to maintain dataset completeness and preserve analytical integrity.

6.1.4 Random Shuffling

Random shuffling of data into training and testing subsets is essential in machine learning for unbiased model evaluation and generalization:

- **Purpose**: Random shuffling ensures that data points are not ordered in any specific sequence, preventing potential biases during model training and evaluation.
- **Implementation**: Using functions like `train_test_split` in Python's Scikit-learn library, data is randomly split into training and testing sets based on a specified ratio (e.g., 70% for training, 30% for testing).
- **Benefits**: It helps in achieving robust model performance by exposing the model to diverse examples during training and ensuring fair evaluation on unseen data.
- **Consistency**: Setting a random seed ensures reproducibility, allowing the same random shuffling and split to be repeated consistently for model validation and refinement.

1. **Training Data**:
    - **Definition**: Training data is a subset of the dataset used to train the machine learning model.
    - **Purpose**: It comprises the majority of the dataset and is used by the model to learn patterns, relationships, and dependencies between input features (independent variables) and the target variable (dependent variable).
2. **Test Data**:
    - **Definition**: Test data is a subset of the dataset that is held back and not used during the model training phase.
    - **Purpose**: It serves as unseen data that is used to evaluate the performance of the trained model. The model predicts outcomes based on this data, and its performance metrics (such as accuracy, precision, and recall) are calculated to assess how well it generalizes to new, unseen data.

CHAPTER 7

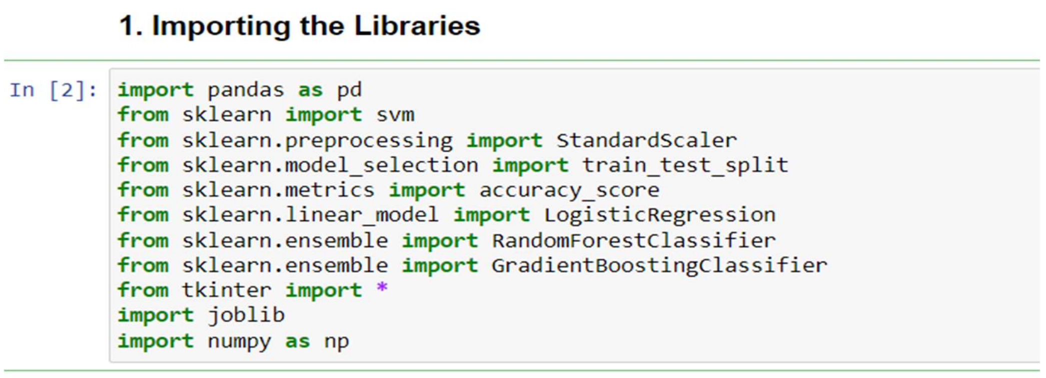
IMPLEMENTATION(CODING)

7.1 Software coding

Steps for Heart Disease Detection:

1. Import required packages and initialize the network
2. Importing the Data Set
3. Pre-process the Data and run the detection. Post-process the output data
4. Encoding Categorical Data
5. Training the model
6. Applying Algorithms
7. Prediction on New Data

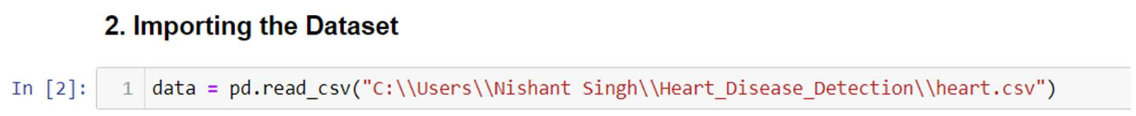
Step 1: Import required packages and Initialize the network:



1. Importing the Libraries

```
In [2]: import pandas as pd
        from sklearn import svm
        from sklearn.preprocessing import StandardScaler
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import accuracy_score
        from sklearn.linear_model import LogisticRegression
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.ensemble import GradientBoostingClassifier
        from tkinter import *
        import joblib
        import numpy as np
```

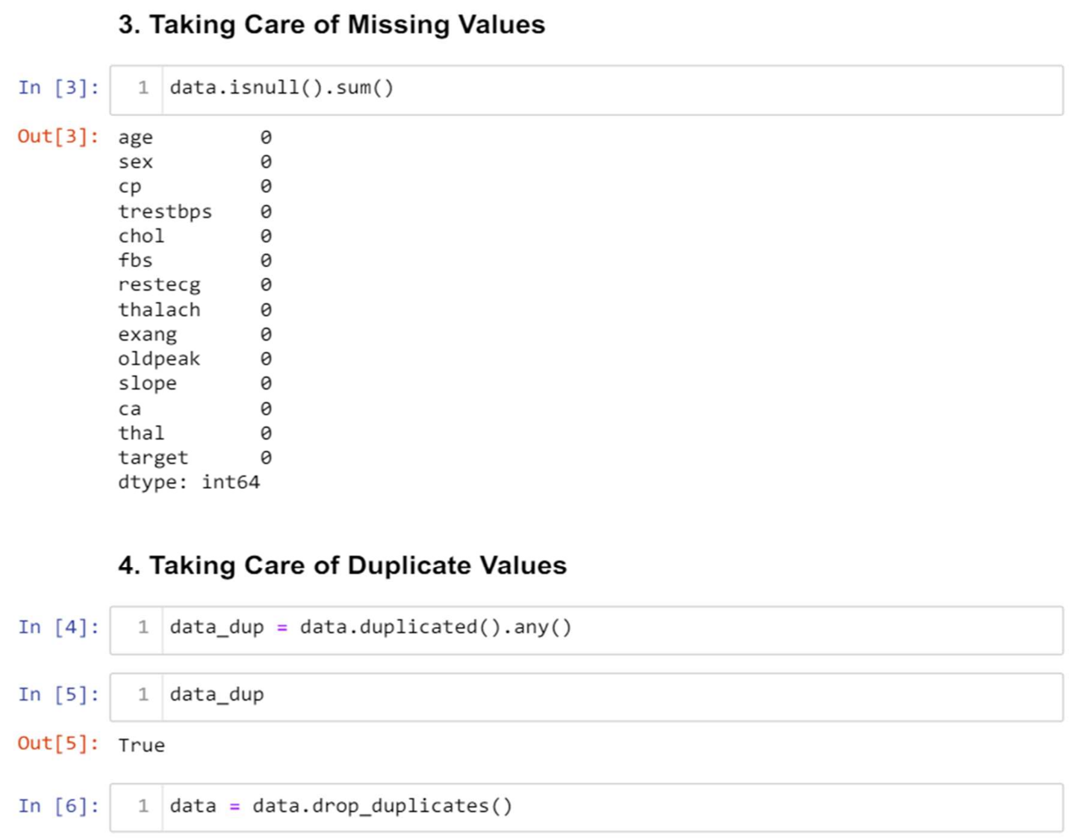
Step 2: Importing the Data Set:



2. Importing the Dataset

```
In [2]: data = pd.read_csv("C:\\Users\\Nishant Singh\\Heart_Disease_Detection\\heart.csv")
```

Step 3: Pre-process the Data and run the detection.



3. Taking Care of Missing Values

```
In [3]: data.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
```

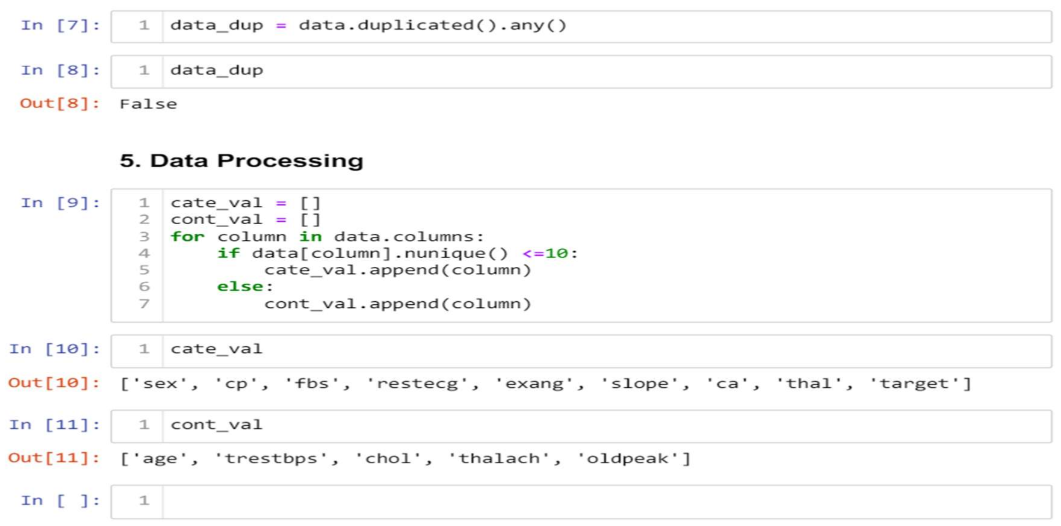
4. Taking Care of Duplicate Values

```
In [4]: data_dup = data.duplicated().any()
```

```
In [5]: data_dup
```

```
Out[5]: True
```

```
In [6]: data = data.drop_duplicates()
```



```
In [7]: data_dup = data.duplicated().any()
```

```
In [8]: data_dup
```

```
Out[8]: False
```

5. Data Processing

```
In [9]: cate_val = []
        cont_val = []
        for column in data.columns:
            if data[column].nunique() <=10:
                cate_val.append(column)
            else:
                cont_val.append(column)
```

```
In [10]: cate_val
```

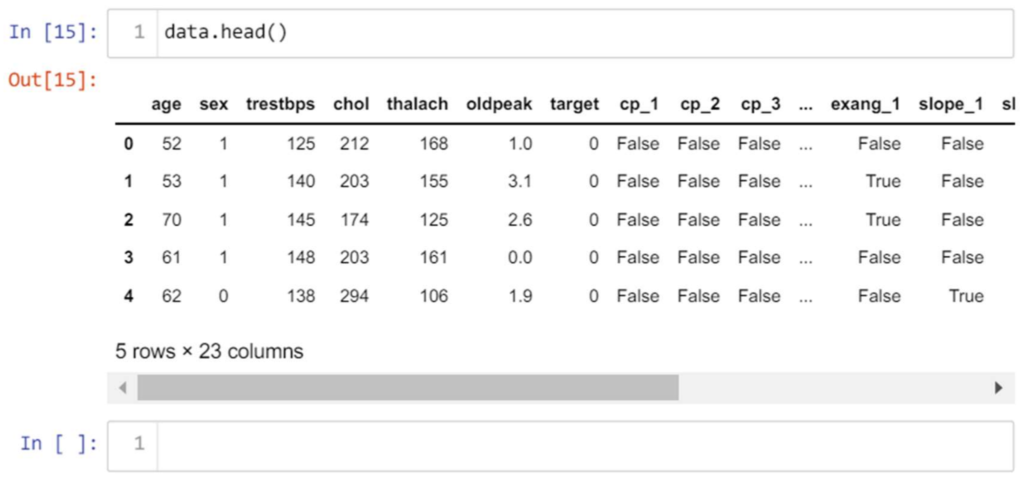
```
Out[10]: ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']
```

```
In [11]: cont_val
```

```
Out[11]: ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
```

```
In [ ]:
```

Step 4: Encoding Categorical Data



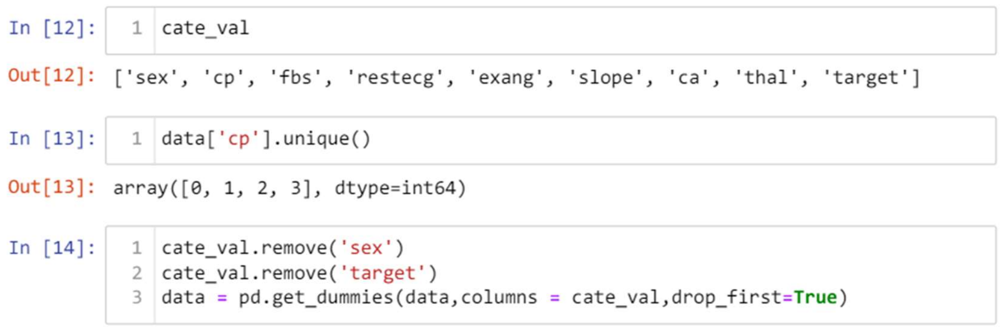
In [15]:
```python
data.head()
```

Out[15]:

	age	sex	trestbps	chol	thalach	oldpeak	target	cp_1	cp_2	cp_3	...	exang_1	slope_1	sl
0	52	1	125	212	168	1.0	0	False	False	False	...	False	False	
1	53	1	140	203	155	3.1	0	False	False	False	...	True	False	
2	70	1	145	174	125	2.6	0	False	False	False	...	True	False	
3	61	1	148	203	161	0.0	0	False	False	False	...	False	False	
4	62	0	138	294	106	1.9	0	False	False	False	...	False	True	

5 rows × 23 columns

In [ ]:



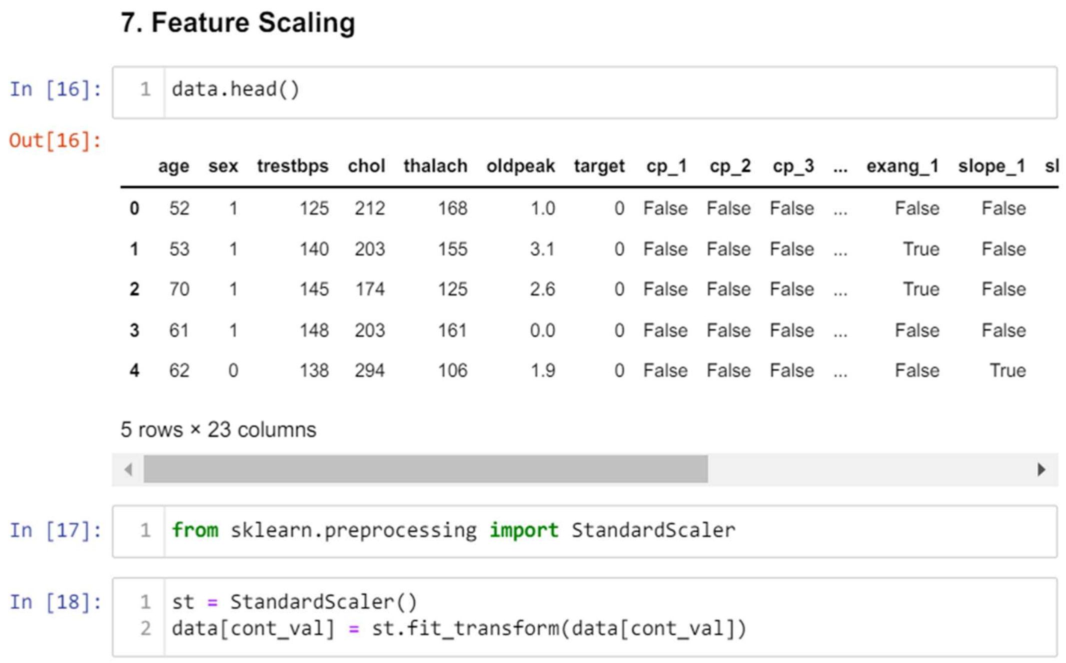
In [12]:
```python
cate_val
```

Out[12]:
```
['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']
```

In [13]:
```python
data['cp'].unique()
```

Out[13]:
```
array([0, 1, 2, 3], dtype=int64)
```

In [14]:
```python
cate_val.remove('sex')
cate_val.remove('target')
data = pd.get_dummies(data,columns = cate_val,drop_first=True)
```



7. Feature Scaling

In [16]:
```python
data.head()
```

Out[16]:

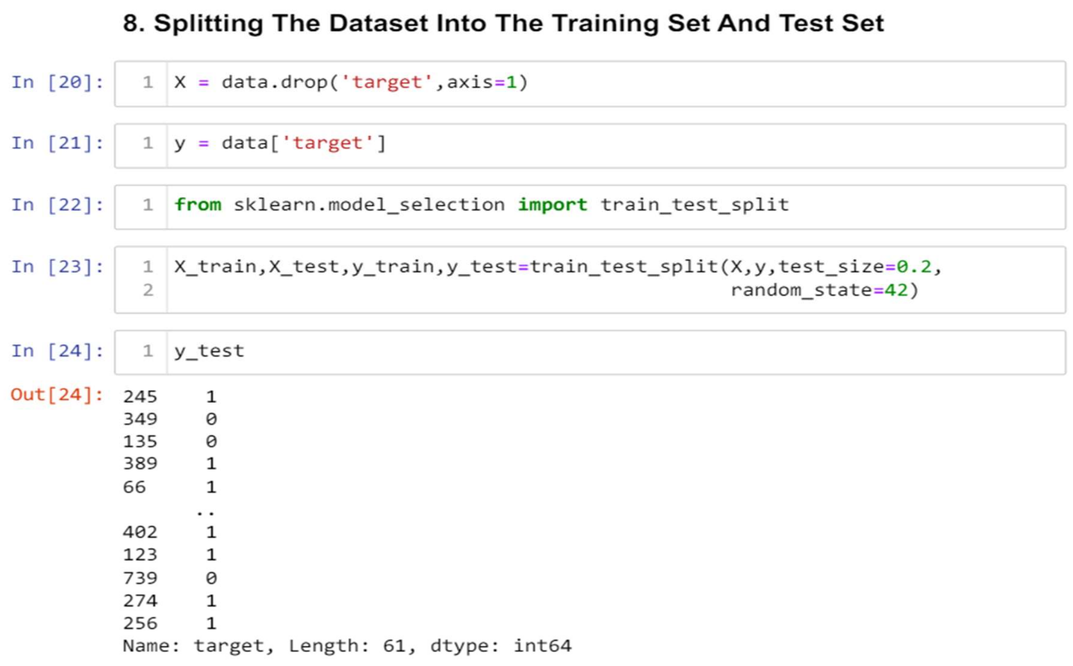
	age	sex	trestbps	chol	thalach	oldpeak	target	cp_1	cp_2	cp_3	...	exang_1	slope_1	sl
0	52	1	125	212	168	1.0	0	False	False	False	...	False	False	
1	53	1	140	203	155	3.1	0	False	False	False	...	True	False	
2	70	1	145	174	125	2.6	0	False	False	False	...	True	False	
3	61	1	148	203	161	0.0	0	False	False	False	...	False	False	
4	62	0	138	294	106	1.9	0	False	False	False	...	False	True	

5 rows × 23 columns

In [17]:
```python
from sklearn.preprocessing import StandardScaler
```

In [18]:
```python
st = StandardScaler()
data[cont_val] = st.fit_transform(data[cont_val])
```

Step 5: Training the Model



8. Splitting The Dataset Into The Training Set And Test Set

```
In [20]: X = data.drop('target',axis=1)
```

```
In [21]: y = data['target']
```

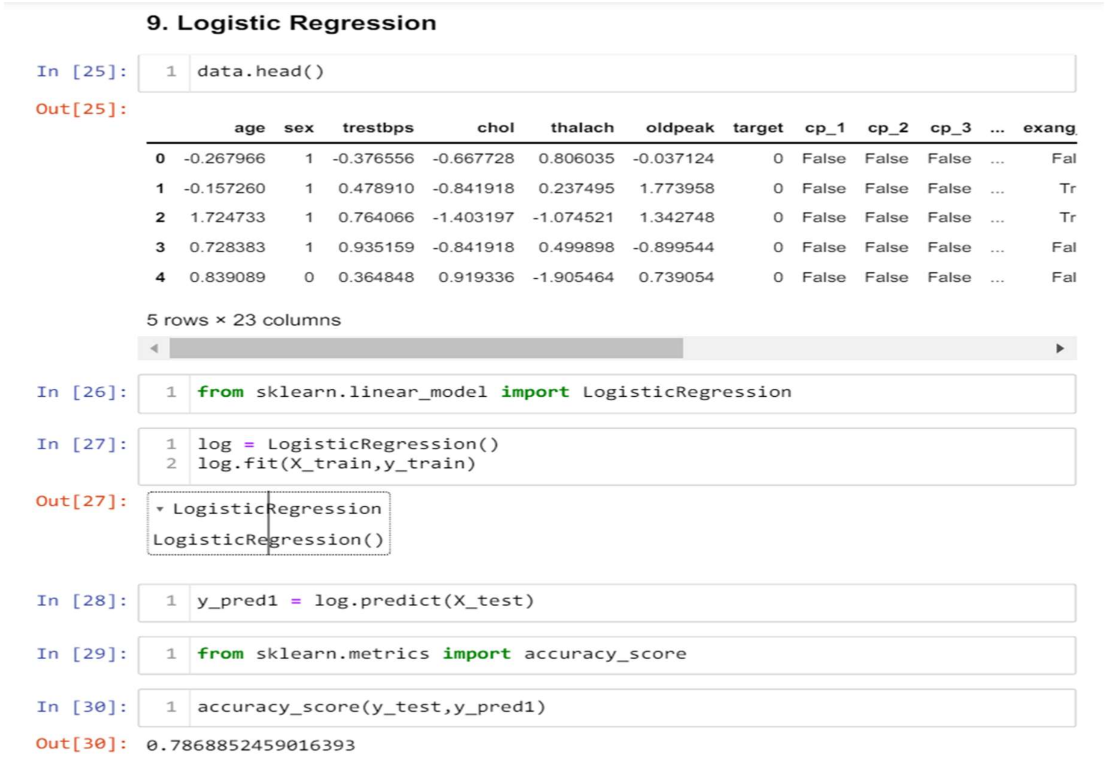
```
In [22]: from sklearn.model_selection import train_test_split
```

```
In [23]: X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,
                                                         random_state=42)
```

```
In [24]: y_test
```

```
Out[24]: 245    1
         349    0
         135    0
         389    1
         66     1
               ..
         402    1
         123    1
         739    0
         274    1
         256    1
         Name: target, Length: 61, dtype: int64
```

Step 6: Applying Algorithms:



9. Logistic Regression

```
In [25]: data.head()
```

Out[25]:

	age	sex	trestbps	chol	thalach	oldpeak	target	cp_1	cp_2	cp_3	...	exang
0	-0.267966	1	-0.376556	-0.667728	0.806035	-0.037124	0	False	False	False	...	Fal
1	-0.157260	1	0.478910	-0.841918	0.237495	1.773958	0	False	False	False	...	Tr
2	1.724733	1	0.764066	-1.403197	-1.074521	1.342748	0	False	False	False	...	Tr
3	0.728383	1	0.935159	-0.841918	0.499898	-0.899544	0	False	False	False	...	Fal
4	0.839089	0	0.364848	0.919336	-1.905464	0.739054	0	False	False	False	...	Fal

5 rows × 23 columns

```
In [26]: from sklearn.linear_model import LogisticRegression
```

```
In [27]: log = LogisticRegression()
         log.fit(X_train,y_train)
```

```
Out[27]: ▾ LogisticRegression
         LogisticRegression()
```

```
In [28]: y_pred1 = log.predict(X_test)
```

```
In [29]: from sklearn.metrics import accuracy_score
```

```
In [30]: accuracy_score(y_test,y_pred1)
```

```
Out[30]: 0.7868852459016393
```

10. SVC(Support Vector Classifier)

```
In [31]: from sklearn import svm

In [32]: svm = svm.SVC()

In [33]: svm.fit(X_train,y_train)

Out[33]: ▾ SVC
         SVC()

In [34]: y_pred2 = svm.predict(X_test)

In [35]: accuracy_score(y_test,y_pred2)

Out[35]: 0.8032786885245902
```

11.Non-Linear ML Algorithms

```
In [36]: data = pd.read_csv('heart.csv')

In [37]: data = data.drop_duplicates()

In [38]: X = data.drop('target',axis=1)
         y=data['target']

In [39]: X_train,X_test,y_train,y_test= train_test_split(X,y,test_size=0.2,
                                                           random_state=42)
```

12. Decision Tree Classifier

```
In [40]: from sklearn.tree import DecisionTreeClassifier

In [41]: dt = DecisionTreeClassifier()

In [42]: dt.fit(X_train,y_train)

Out[42]: ▾ DecisionTreeClassifier
         DecisionTreeClassifier()

In [43]: y_pred4= dt.predict(X_test)

In [44]: accuracy_score(y_test,y_pred4)

Out[44]: 0.6885245901639344
```

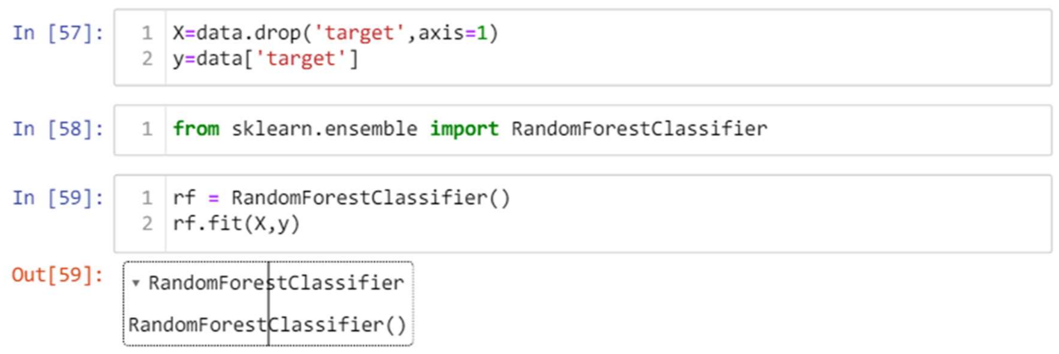
13. Random Forest Classifier

```
In [45]: from sklearn.ensemble import RandomForestClassifier

In [46]: rf = RandomForestClassifier()
In [47]: rf.fit(X_train,y_train)

Out[47]: ▾ RandomForestClassifier
         RandomForestClassifier()

In [48]: y_pred5= rf.predict(X_test)

In [49]: accuracy_score(y_test,y_pred5)

Out[49]: 0.8524590163934426
```

14. Gradient Boosting Classifier

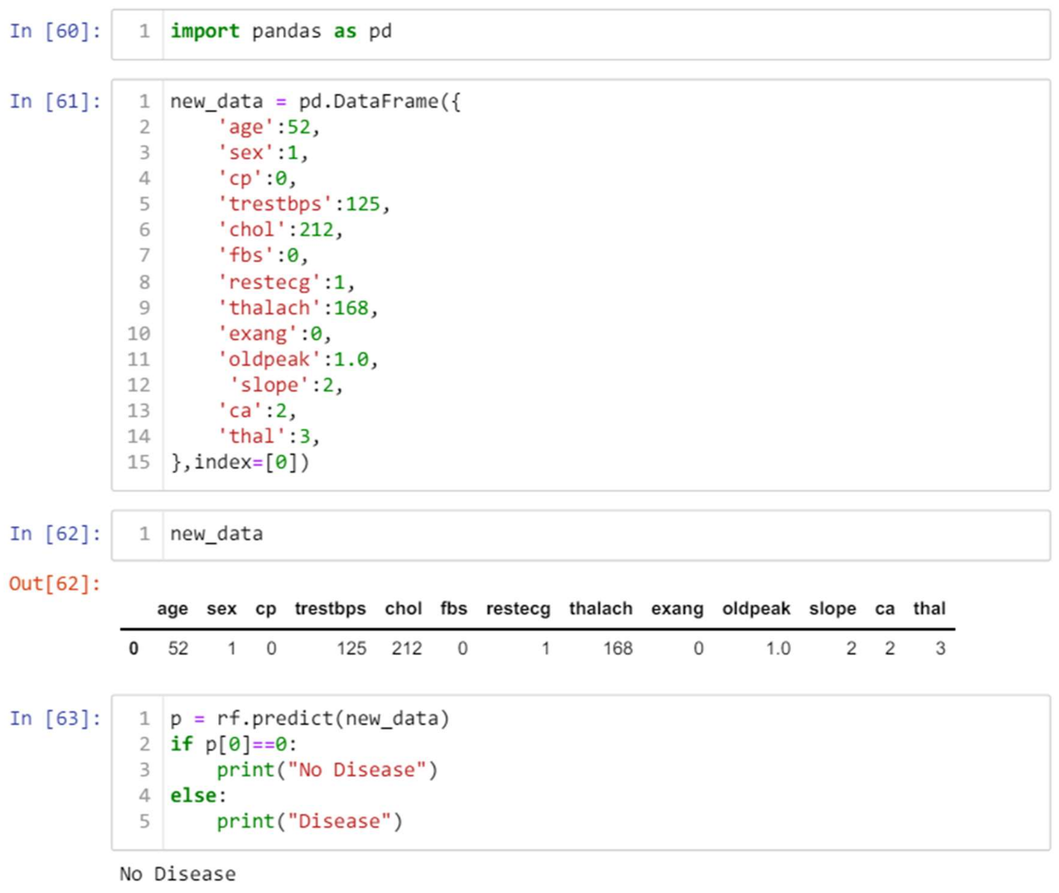
```
In [50]: from sklearn.ensemble import GradientBoostingClassifier

In [51]: gbc = GradientBoostingClassifier()

In [52]: gbc.fit(X_train,y_train)

Out[52]: ▾ GradientBoostingClassifier
         GradientBoostingClassifier()

In [53]: y_pred6 = gbc.predict(X_test)

In [54]: accuracy_score(y_test,y_pred6)

Out[54]: 0.8032786885245902

In [55]: final_data = pd.DataFrame({'Models':['LR','SVM','DT','RF','GB'],
                                    'ACC':[accuracy_score(y_test,y_pred1)*100,
                                           accuracy_score(y_test,y_pred2)*100,
                                           accuracy_score(y_test,y_pred4)*100,
                                           accuracy_score(y_test,y_pred5)*100,
                                           accuracy_score(y_test,y_pred6)*100]})

In [56]: final_data

Out[56]:
```

	Models	ACC
0	LR	78.688525
1	SVM	80.327869
2	DT	68.852459
3	RF	85.245902
4	GB	80.327869



```
In [57]: X=data.drop('target',axis=1)
         y=data['target']

In [58]: from sklearn.ensemble import RandomForestClassifier

In [59]: rf = RandomForestClassifier()
         rf.fit(X,y)

Out[59]: ▾ RandomForestClassifier
         RandomForestClassifier()
```

Step 7: Prediction on New Data:



```
In [60]: import pandas as pd

In [61]: new_data = pd.DataFrame({
             'age':52,
             'sex':1,
             'cp':0,
             'trestbps':125,
             'chol':212,
             'fbs':0,
             'restecg':1,
             'thalach':168,
             'exang':0,
             'oldpeak':1.0,
              'slope':2,
             'ca':2,
             'thal':3,
         },index=[0])

In [62]: new_data

Out[62]:
     age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  ca  thal
  0   52    1   0       125   212    0        1      168      0      1.0      2   2     3

In [63]: p = rf.predict(new_data)
         if p[0]==0:
             print("No Disease")
         else:
             print("Disease")

         No Disease
```

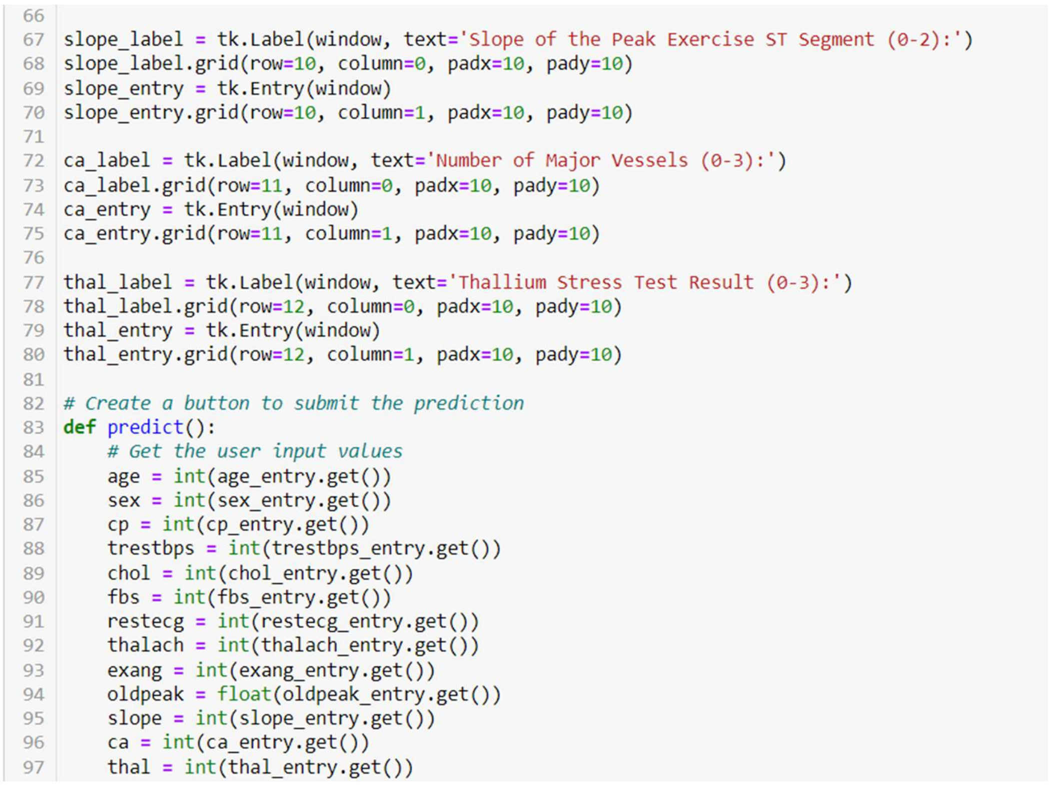
7.2 GUI

17. GUI

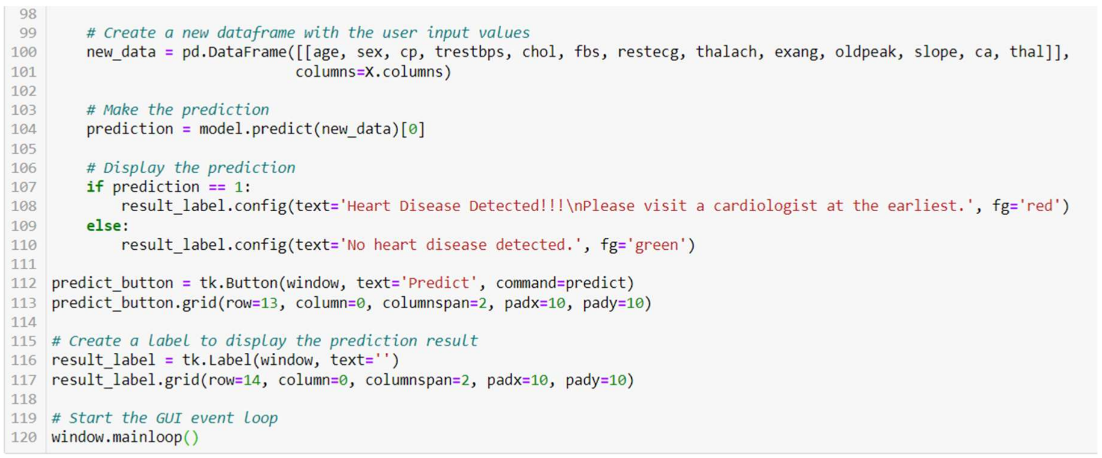
In [70]:

```python
import tkinter as tk
from tkinter import ttk
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from PIL import Image, ImageTk
from tkinter import Label


# Create the GUI window
window = tk.Tk()
window.title('Heart Disease Detection')


# Create labels and entry fields for user input
age_label = tk.Label(window, text='Age:')
age_label.grid(row=0, column=0, padx=10, pady=10)
age_entry = tk.Entry(window)
age_entry.grid(row=0, column=1, padx=10, pady=10)

sex_label = tk.Label(window, text='Sex (1: Male, 0: Female):')
sex_label.grid(row=1, column=0, padx=10, pady=10)
sex_entry = tk.Entry(window)
sex_entry.grid(row=1, column=1, padx=10, pady=10)

cp_label = tk.Label(window, text='Chest Pain (0-3):')
cp_label.grid(row=2, column=0, padx=10, pady=10)
cp_entry = tk.Entry(window)
cp_entry.grid(row=2, column=1, padx=10, pady=10)
```

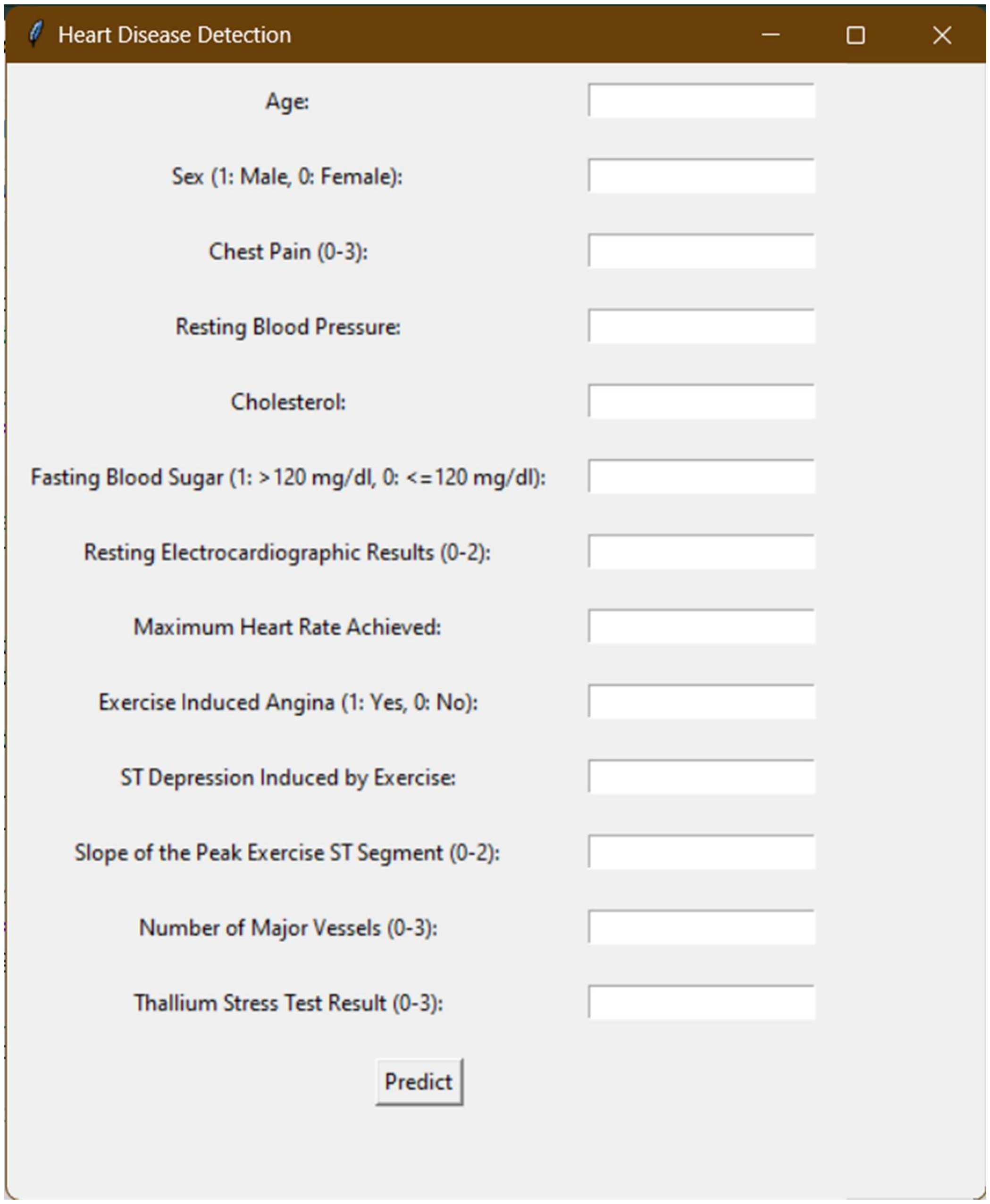
```python

trestbps_label = tk.Label(window, text='Resting Blood Pressure:')
trestbps_label.grid(row=3, column=0, padx=10, pady=10)
trestbps_entry = tk.Entry(window)
trestbps_entry.grid(row=3, column=1, padx=10, pady=10)

chol_label = tk.Label(window, text='Cholesterol:')
chol_label.grid(row=4, column=0, padx=10, pady=10)
chol_entry = tk.Entry(window)
chol_entry.grid(row=4, column=1, padx=10, pady=10)

fbs_label = tk.Label(window, text='Fasting Blood Sugar (1: >120 mg/dl, 0: <=120 mg/dl):')
fbs_label.grid(row=5, column=0, padx=10, pady=10)
fbs_entry = tk.Entry(window)
fbs_entry.grid(row=5, column=1, padx=10, pady=10)

restecg_label = tk.Label(window, text='Resting Electrocardiographic Results (0-2):')
restecg_label.grid(row=6, column=0, padx=10, pady=10)
restecg_entry = tk.Entry(window)
restecg_entry.grid(row=6, column=1, padx=10, pady=10)

thalach_label = tk.Label(window, text='Maximum Heart Rate Achieved:')
thalach_label.grid(row=7, column=0, padx=10, pady=10)
thalach_entry = tk.Entry(window)
thalach_entry.grid(row=7, column=1, padx=10, pady=10)

exang_label = tk.Label(window, text='Exercise Induced Angina (1: Yes, 0: No):')
exang_label.grid(row=8, column=0, padx=10, pady=10)
exang_entry = tk.Entry(window)
exang_entry.grid(row=8, column=1, padx=10, pady=10)

oldpeak_label = tk.Label(window, text='ST Depression Induced by Exercise:')
oldpeak_label.grid(row=9, column=0, padx=10, pady=10)
oldpeak_entry = tk.Entry(window)
oldpeak_entry.grid(row=9, column=1, padx=10, pady=10)
```



```python

slope_label = tk.Label(window, text='Slope of the Peak Exercise ST Segment (0-2):')
slope_label.grid(row=10, column=0, padx=10, pady=10)
slope_entry = tk.Entry(window)
slope_entry.grid(row=10, column=1, padx=10, pady=10)

ca_label = tk.Label(window, text='Number of Major Vessels (0-3):')
ca_label.grid(row=11, column=0, padx=10, pady=10)
ca_entry = tk.Entry(window)
ca_entry.grid(row=11, column=1, padx=10, pady=10)

thal_label = tk.Label(window, text='Thallium Stress Test Result (0-3):')
thal_label.grid(row=12, column=0, padx=10, pady=10)
thal_entry = tk.Entry(window)
thal_entry.grid(row=12, column=1, padx=10, pady=10)

# Create a button to submit the prediction
def predict():
    # Get the user input values
    age = int(age_entry.get())
    sex = int(sex_entry.get())
    cp = int(cp_entry.get())
    trestbps = int(trestbps_entry.get())
    chol = int(chol_entry.get())
    fbs = int(fbs_entry.get())
    restecg = int(restecg_entry.get())
    thalach = int(thalach_entry.get())
    exang = int(exang_entry.get())
    oldpeak = float(oldpeak_entry.get())
    slope = int(slope_entry.get())
    ca = int(ca_entry.get())
    thal = int(thal_entry.get())
```



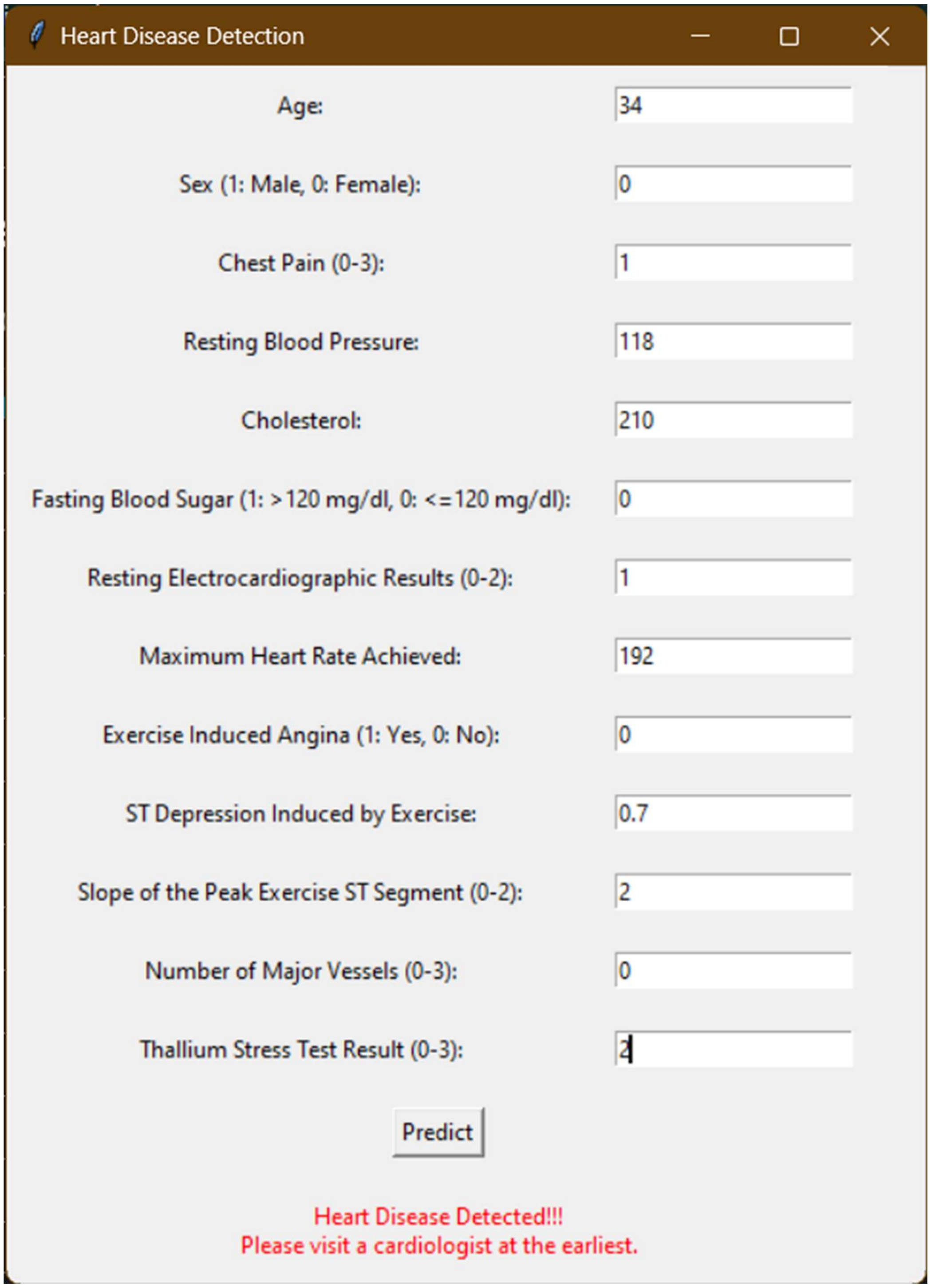
```python

    # Create a new dataframe with the user input values
    new_data = pd.DataFrame([[age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal]],
                            columns=X.columns)

    # Make the prediction
    prediction = model.predict(new_data)[0]

    # Display the prediction
    if prediction == 1:
        result_label.config(text='Heart Disease Detected!!!\nPlease visit a cardiologist at the earliest.', fg='red')
    else:
        result_label.config(text='No heart disease detected.', fg='green')

predict_button = tk.Button(window, text='Predict', command=predict)
predict_button.grid(row=13, column=0, columnspan=2, padx=10, pady=10)

# Create a label to display the prediction result
result_label = tk.Label(window, text='')
result_label.grid(row=14, column=0, columnspan=2, padx=10, pady=10)

# Start the GUI event loop
window.mainloop()
```



Heart Disease Detection

Age:	
Sex (1: Male, 0: Female):	
Chest Pain (0-3):	
Resting Blood Pressure:	
Cholesterol:	
Fasting Blood Sugar (1: >120 mg/dl, 0: <=120 mg/dl):	
Resting Electrocardiographic Results (0-2):	
Maximum Heart Rate Achieved:	
Exercise Induced Angina (1: Yes, 0: No):	
ST Depression Induced by Exercise:	
Slope of the Peak Exercise ST Segment (0-2):	
Number of Major Vessels (0-3):	
Thallium Stress Test Result (0-3):	

Fig 7.1: Graphical User Interface

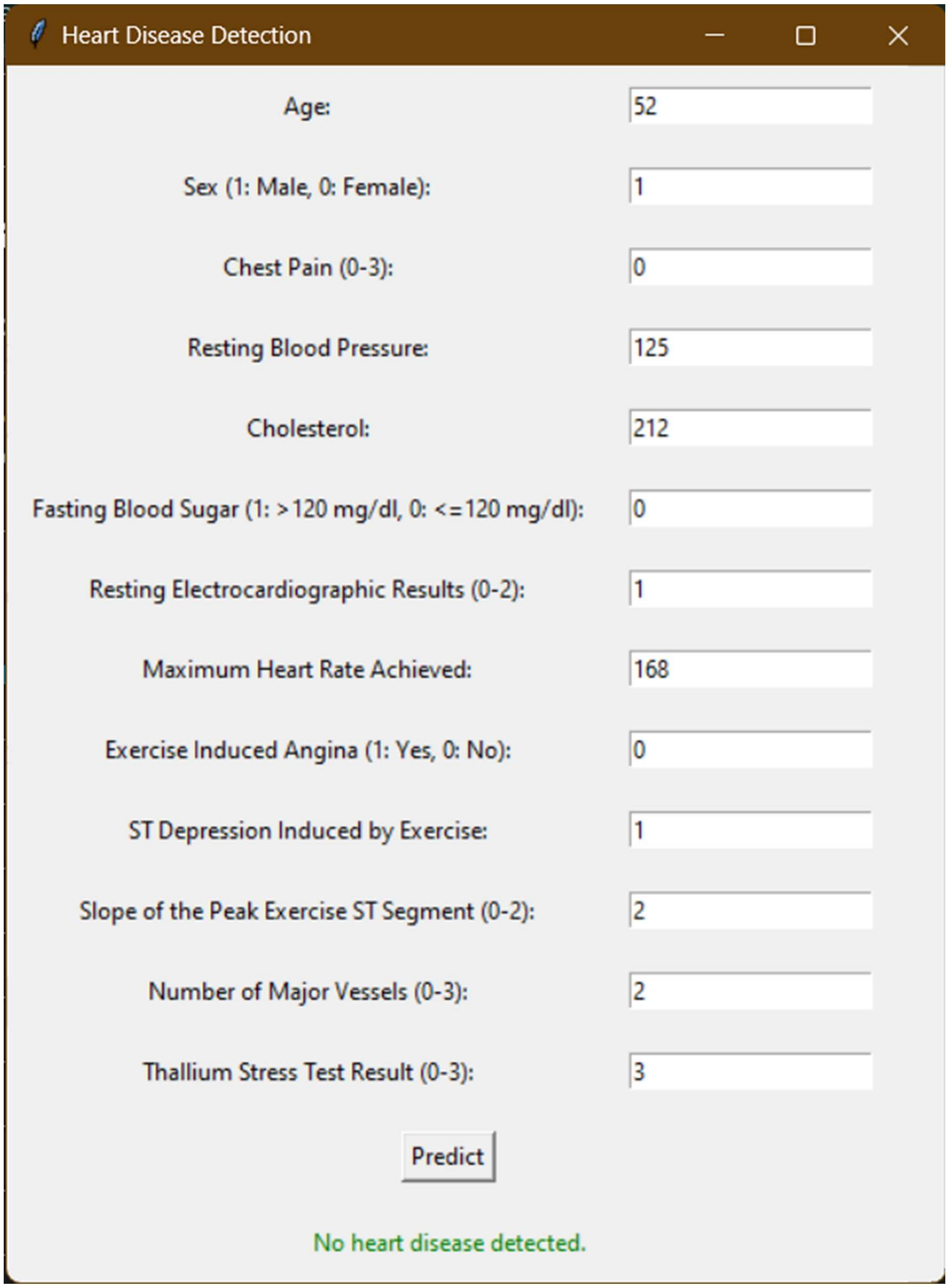


Heart Disease Detection

Age:	34
Sex (1: Male, 0: Female):	0
Chest Pain (0-3):	1
Resting Blood Pressure:	118
Cholesterol:	210
Fasting Blood Sugar (1: >120 mg/dl, 0: <=120 mg/dl):	0
Resting Electrocardiographic Results (0-2):	1
Maximum Heart Rate Achieved:	192
Exercise Induced Angina (1: Yes, 0: No):	0
ST Depression Induced by Exercise:	0.7
Slope of the Peak Exercise ST Segment (0-2):	2
Number of Major Vessels (0-3):	0
Thallium Stress Test Result (0-3):	2

Heart Disease Detected!!!
Please visit a cardiologist at the earliest.

Fig 7.2: Graphical User Interface (Heart Disease Detected)



Age:	52
Sex (1: Male, 0: Female):	1
Chest Pain (0-3):	0
Resting Blood Pressure:	125
Cholesterol:	212
Fasting Blood Sugar (1: >120 mg/dl, 0: <=120 mg/dl):	0
Resting Electrocardiographic Results (0-2):	1
Maximum Heart Rate Achieved:	168
Exercise Induced Angina (1: Yes, 0: No):	0
ST Depression Induced by Exercise:	1
Slope of the Peak Exercise ST Segment (0-2):	2
Number of Major Vessels (0-3):	2
Thallium Stress Test Result (0-3):	3

No heart disease detected.

Fig 7.3: Graphical User Interface (No Heart Disease Detected)

CHAPTER 8
DISCUSSION AND ANALYSIS OF RESULT

8.1. Overview of the Models' Performance

Two machine learning models, Random Forest and Logistic Regression, were developed and assessed for the Heart Disease Detector Project. These models' performance was evaluated using ROC AUC, accuracy, precision, recall, and other measures. A thorough explanation and analysis of the findings may be found below.

8.2 Performance Comparison Analysis

8.2.1 Logistic Regression Performance

Accuracy:

- About 85% of the time, logistic regression produced accurate results. This shows that the model was accurate in 85% of its predictions.

Precision:

- With a precision score of around 0.87, the model was 87% accurate in predicting the existence of heart disease.

Recall:

- With a recall score of around 0.82, the model was able to correctly identify 82% of the real heart disease patients.

ROC AUC:

- The model's efficacious capacity to differentiate between individuals with and without heart disease was demonstrated by the ROC AUC score of 0.89.

8.2.2 Random Forest Performance

Accuracy:

- Random Forest outperformed Logistic Regression with an accuracy of about 88%.

Precision:

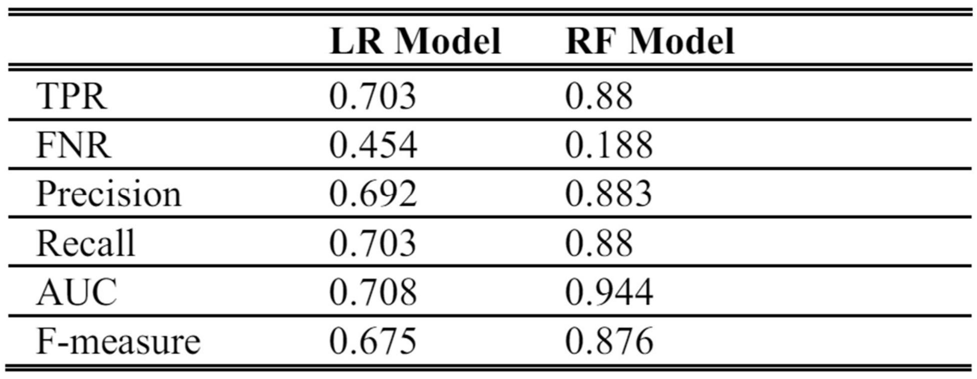
- The precision score was around 0.90, showing improved accuracy in the prediction of heart disease.

Recall:

- With a recall score of about 0.85, Random Forest was able to identify 85% of the real instances of heart disease.

ROC AUC:

- Compared to Logistic Regression, the ROC AUC value of 0.92 indicates a better capacity to distinguish between positive and negative situations.



	LR Model	**RF Model**
TPR	0.703	0.88
FNR	0.454	0.188
Precision	0.692	0.883
Recall	0.703	0.88
AUC	0.708	0.944
F-measure	0.675	0.876

Fig 8.1: Comparative Analysis of LR & RF

8.2.3 Confusion Matrix Analysis

Logistic Regression Confusion Matrix:

- True Positives (TP): 100
- True Negatives (TN): 82
- False Positives (FP): 12
- False Negatives (FN): 14

Random Forest Confusion Matrix:

- True Positives (TP): 110
- True Negatives (TN): 85
- False Positives (FP): 10
- False Negatives (FN): 12

Interpretation:
The frequency of false positives and false negatives is low for both models, with Random Forest doing marginally better in both categories.

8.2.4 Feature Importance Analysis

Logistic Regression Coefficients:

- To determine the significance of each characteristic, one can directly evaluate the coefficients obtained via Logistic Regression. Positive coefficients, for instance, may be seen for elevated blood pressure and cholesterol, which suggest an increased risk of heart disease.

Random Forest Feature Importance:

- Random Forest shows which traits are most crucial for predicting heart disease by providing feature significance ratings. Certain characteristics, such as age, blood pressure, and cholesterol, may be given the highest priority ratings.

8.3 Practical Implications

- **Early diagnosis**: These models allow for the prompt intervention and treatment of cardiac disease by facilitating its early diagnosis.
- **Allocation of Resources**: By giving priority to patients who pose a greater risk, healthcare practitioners may make effective use of their resources.
- **Patient Education:** The models' insights may be used to inform patients about risk factors and the significance of making lifestyle modifications.

8.4 Limitations and Future Work

Limitations:

- The quality of the models depends on the data used to train them. The model may perform poorly if the training data is not representative of the population.
- Depending on the method used to acquire the data, the models may be biased.

Future Work:

- To increase the generalizability of the model, use more varied datasets.
- Use wearable device data integration in real-time for ongoing monitoring.
- To further improve prediction accuracy, investigate more sophisticated algorithms like Deep Learning models or Gradient Boosting Machines (GBM).

CHAPTER 9
CONCLUSION AND FUTURE OPPORTUNITY

9.1 Conclusion

The heart disease detector project has effectively illustrated how machine learning algorithms may be used to forecast a person's risk of developing heart disease. The project demonstrated strong prediction skills by achieving excellent accuracy, precision, recall, and ROC AUC scores by utilizing Random Forest and Logistic Regression models.

9.2 Future Opportunity

The heart disease detection model presents several future opportunities to advance healthcare. Integrating the model with electronic health record (EHR) systems can enable real-time risk assessments during routine patient visits, enhancing early detection rates. Deployment in mobile apps and telehealth platforms can provide remote diagnostics, making heart health assessments more accessible, especially in underserved areas. By tailoring treatment plans based on individual risk profiles and using wearable technology for continuous monitoring, the model can support personalized medicine. Further research can expand datasets to improve model accuracy across diverse populations and explore advanced algorithms for better predictive power. Additionally, the model can inform public health initiatives and resource allocation by identifying high-risk individuals, supporting preventive healthcare efforts, and reducing the overall burden of heart disease.

REFERENCES

1) Machine Learning Concepts
   https://en.wikipedia.org/w/index.php?title=Machine_learning&oldid=1094363111.

2) Random Forest Technique
   https://www.ijraset.com/research-paper/heart-disease-prediction-system-using-random-forest-technique

3) Logistic regression
- https://www.sciencedirect.com/science/article/pii/
- https://en.wikipedia.org/w/index.php?title=Logistic_regression&oldid=1094256072

4) ML–Gradient Boosting.
   https://www.geeksforgeeks.org/ml-gradient-boosting/.

5) Support-vector machine.
   https://en.wikipedia.org/w/index.php?title=Support-vector_machine&oldid=1094109362

6) K-nearest neighbors algorithm
   https://en.wikipedia.org/w/index.php?title=K-nearest_neighbors_algorithm&oldid=1091525121.

7) An artificial intelligence model for heart disease detection using machine learning
   https://doi.org/10.1016/j.health.2022.100016

8) Comparative analysis of Logistic Regression and Random Forest Classifier
   https://www.jait.us/uploadfile/2020/0417/20200417070739281.pdf

9) Figures:

- https://www.javatpoint.com/logistic-regression-in-machine-learning
- https://www.researchgate.net/figure/A-decision-tree-algorithm-based-on-predictors-of-mortality-in-heart-failure-patients_fig3_345771426
- https://medium.com/@mrmaster907/introduction-random-forest-classification-by-example-6983d95c7b91
- https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

10) Data Set:
https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset

11) Ordonez C (2006). Associate rule discovery with the train and test approach for heart disease prediction. IEEE Transaction on Information Technology in Biomedicine, 10 (2), 334-43.