CHAPTER 1

INTRODUCTION

Globally, cardiovascular diseases (CVDs) constitute the primary cause of mortality. Among them, myocardial infarction (MI), sometimes referred to as a heart attack, is one of the most serious ailments. A myocardial infarction happens when a blood clot or other blockage cuts off the blood supply to a portion of the heart muscle, causing damage or even death to the heart tissue. For prompt medical intervention to enhance patient outcomes and survival rates, early identification and precise prediction of myocardial infarction are critical. The conventional method of diagnosing myocardial infarction combines electrocardiograms (ECGs), biomarkers, physical examinations, and patient histories with clinical evaluations. Although these techniques work well, they frequently have drawbacks in terms of how quickly and thoroughly the diagnosis can be made. The necessity for real-time risk assessment and the growing complexity of medical data have brought attention to the advantages of incorporating cutting-edge analytical methods like machine learning into the diagnostic process.

1.1 Motivation

Myocardial infarction prediction incorporates machine learning to handle the problem of changing healthcare demands, which goes beyond immediate clinical gains. Traditional procedures by themselves are not enough to adequately address this expanding dilemma as the worldwide burden of cardiovascular illnesses keeps rising. To manage massive data sets and identify minute danger indicators that more conventional methods could miss, machine learning offers a potent tool. By detecting at-risk patients before symptoms appear, this capacity not only improves diagnostic precision but also promotes proactive healthcare. To ensure that prediction models stay current and accurate over time, machine learning also can continually adapt to new data and developing patterns. This strategy has the potential to revolutionise cardiovascular care, spur innovation in medical research, and eventually improve patient quality of life globally by expanding prediction skills and promoting personalised medication.

1.2 About The Disease

When blood supply to a portion of the heart muscle is obstructed, usually due to a clot in the coronary arteries, the result is a heart attack, also known as a myocardial infarction. This disruption in blood flow causes the heart muscle to get damaged or perhaps die. The afflicted cardiac tissue may sustain irreversible damage if blood flow is not quickly restored, which might result in serious problems or even death.

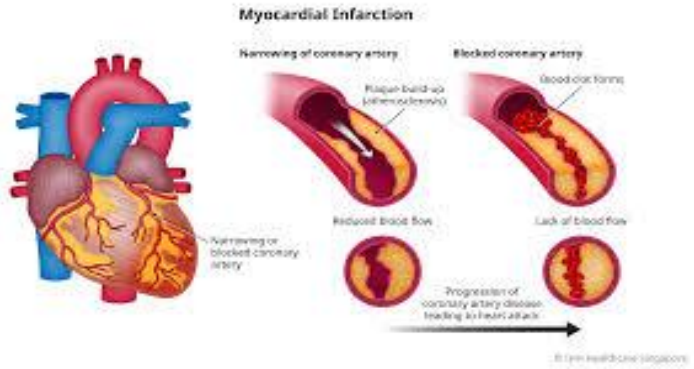

Fig 1.1

Heart attacks are a major cause of death in India, raising serious concerns for public health. The growing incidence of cardiovascular illnesses can be attributed to causes such as rapid urbanization, changing lifestyles, and the rising prevalence of risk factors including obesity, diabetes, and hypertension.Heart attacks are one of the main causes of death worldwide. with the highest fatality rate among cardiovascular disorders. The abrupt nature of heart attacks and the urgent need for medical attention to restore blood flow and prevent severe heart muscle damage are major contributing factors to this high death rate.

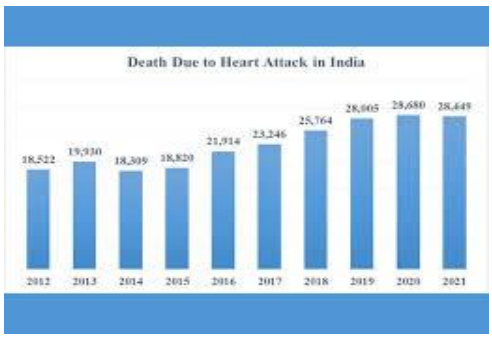

Fig 1.2

HIGHEST MORTALITY

DISEASE	2018	2019	2020	2021*
Heart attack	8,601	5,849	5,633	17,880
Cancer	10,073	9,958	8,576	6,861
Kidney failure	1,396	1,516	1,634	1,182
Covid-19	0	0	11,105	10,289
Tuberculosis	4,940	4,899	3,719	2,921
Head injury	1,021	1,000	760	545

Source: RTI response from BMC *(Jan-June)*

Fig 1.3

1.3 Existing System Overview

Current systems for the diagnosis of myocardial infarction mostly depend on conventional approaches including clinical assessments, electrocardiograms (ECGs), blood tests, and imaging modalities like coronary angiography and echocardiography. Although these methods are fundamental and efficacious, they have several drawbacks, such as inconsistent test findings, delays in diagnosis, and difficulties in combining various data sources. Clinical evaluations may be arbitrary, and conventional diagnostic methods might not fully account for all the variables or combinations of variables that raise the risk of myocardial infarction. Furthermore, these approaches could not make the most of the enormous quantity of patient data that is accessible and frequently call for manual interpretation. As a result, the demand for cutting-edge solutions that can deliver evaluations that are more rapid, precise, and customised is becoming increasingly apparent. Through the analysis of intricate datasets, the discovery of patterns that may go unnoticed, and the provision of real-time risk forecasts to facilitate early intervention and improved patient outcomes, machine learning offers a chance to improve these conventional systems.

1.4 Proposed System Overview

The suggested approach integrates machine learning algorithms into the diagnostic process with the goal of revolutionising the prediction of myocardial infarction. The suggested system uses advanced data analytics to examine a large dataset of patient attributes, including demographic data, clinical indicators, lifestyle factors, and medical history, in contrast to traditional methods that only use static diagnostic tools and manual assessments. The system can recognise intricate patterns and interactions among risk variables that lead to myocardial infarction by utilising machine learning techniques including logistic regression, decision trees, and neural networks. This method improves diagnostic accuracy and speed by enabling personalised forecasts and real-time risk assessment. As fresh data becomes available, the system will continually learn and adjust, enhancing its prediction ability over time. It will also give medical practitioners an intuitive user interface that allows for easy integration into current workflows and promotes well-informed decision-making. The proposed system seeks to improve patient outcomes and advance the area of cardiovascular healthcare by providing a more accurate, effective, and proactive tool for controlling cardiovascular risk.

CHAPTER 2

LITERATURE SURVEY

2.1 Heart disease prediction using machine learning techniques

Authors: Apurv Garg, Bhartendu Sharma and Rijwan Khan

Journal: IOP Publishing

Summary: To increase diagnostic precision and facilitate early intervention, Apurv Garg, Bhartendu Sharma, and Rijwan Khan's paper, which was published by IOP Publishing, investigates the application of machine learning approaches to predict cardiac disease. The researchers use a variety of machine learning methods, such as logistic regression, decision trees, random forests, and support vector machines, using a dataset that contains patient variables like age, sex, blood pressure, and cholesterol levels. According to their findings, when compared to traditional approaches, ensemble methods such as random forests and support vector machines yield higher accuracy and better overall performance. To improve predicted results, the study highlights how crucial model selection and hyperparameter tweaking.

2.2 A symptomatic heart attack prediction method and exploratory analysis

Authors: Lipika Goel, Rohit Tanwar

Journal: F100 Research

Summary: Lipika Goel and Rohit Tanwar's case study explores the prediction of symptomatic heart attacks by means of a comprehensive exploratory analysis of clinical data. The authors use machine learning techniques like logistic regression and decision trees to construct a prediction model by looking at symptoms like shortness of breath, chest discomfort, and other pertinent health markers. Their study shows that there are strong trends and relationships between heart attack risk and symptoms, indicating that symptom-based models can yield more precise forecasts than conventional techniques.

CHAPTER 3

SYSTEM ANALYSIS

Hardware Specifications

Processor (CPU): For moderate workloads, a contemporary multi-core processor (such as an AMD Ryzen or an Intel i5/i7/i9, for example) should be adequate. A high-performance CPU is useful for training complicated models or processing vast amounts of data.

Memory (RAM): It is advised to have at least 8GB of RAM. 16GB or more can be required for larger datasets or sophisticated models.

Storage: SSD storage Quick data access is essential. When storing datasets and model files, an SSD with a minimum capacity of 256GB is advised. Think about adding more storage (such as network-attached storage or external HDDs) for huge datasets.

Network: Accessing cloud resources when necessary and downloading datasets and updates depend on dependable internet connectivity.

Graphics Processing Unit (GPU): A dedicated GPU is highly recommended for effective deep learning model training. To speed up computations, NVIDIA GPUs with CUDA functionality (such as the GTX 1660, RTX 3060, or above) are frequently utilised.

Backup Solutions: It is crucial to regularly backup your data to avoid losing it. External drives and cloud backups are examples of this.

Software Tools and Libraries

Software Requirements

Python: Installed Python 3.x. Python 3.6 and above are compatible with most libraries and tools.

Web Structure: Used Flask to build and run your web application.

Package Management: To manage libraries, used pip, Python's package installer.

Anaconda: we used it to manage environments and packages.

Version Control: For cooperation and version control.

Libraries and Tools

1. Data Handling and Visualization

- **Pandas**: Pandas is a Python module that offers robust data structures like Data Frames and Series for data analysis and manipulation. This facilitates the handling and study of tabular data by streamlining operations including data transformation, cleaning, and analysis. Pandas is an essential Python library for data science and analytics because of its effective capabilities for handling missing data, combining datasets, and carrying out statistical calculations.
- **NumPy:** Large, multi-dimensional arrays and matrices are supported by NumPy, a core Python module for numerical computation. It offers a collection of mathematical procedures for effectively handling these arrays. NumPy is a basic library for scientific computing and is extensively used in data analysis, machine learning, and other computational activities because of its strong array manipulation, linear algebra, and statistical operations capabilities.
- **Matplotlib:** Plots may be customised using a variety of choices, such as colours, labels, line styles, and typefaces, thanks to Matplotlib's flexibility. Toolkits such as mpl_toolkits allow users to add interactive components, include mathematical equations, and construct intricate multi-plot layouts. In order to render images on several platforms and output formats, including PNG, PDF, and SVG, the library also supports a variety of backends. Matplotlib is a preferred tool for Python data visualisation because of its rich documentation and vibrant community.
- **Seaborn:** Seaborn is a robust and intuitive Python data visualisation toolkit developed on top of Matplotlib. It facilitates the creation of intricate visualisations with a little amount of code by offering a high-level interface for creating eye-catching and educational statistics images. Seaborn seamlessly handles DataFrames and supports categorical data because of its close integration with Pandas data structures. To improve the visuals' aesthetic appeal, the library offers a range of plot styles, including heatmaps, box plots, violin plots, and bar plots. It also comes with pre-installed themes

and colour schemes.

2. Machine Learning

- **Sckit-learn:** A flexible Python machine learning framework, scikit-learn offers a plethora of tools for data mining and analysis. It includes tools for model selection, assessment, and preprocessing in addition to user-friendly implementations of many algorithms for classification, regression, clustering, and dimensionality reduction. It is a well-liked option for creating and assessing machine learning models because of its reliable API and compatibility with other libraries like NumPy and pandas.
- **Xgboost:** For supervised learning tasks, XGBoost, also known as eXtreme Gradient Boosting, is a potent and effective machine learning package. It uses a scalable gradient-boosting approach that has excellent accuracy and can handle big datasets. Regression, classification, and ranking are just a few of the goal functions that XGBoost offers. It also works nicely with Python and R. The management of missing data, parallel processing for quicker calculation, and regularisation to avoid overfitting are important characteristics. XGBoost is a well-liked option for both production systems and contests because to its resilience and adaptability.

3. WEB FRAME WORK

- **Html:** The standard language used to generate and design webpages is called HTML, or Hypertext Markup Language. It gives a webpage its structure by defining the content using a few components and tags, including headers, paragraphs, links, photos, and other multimedia. The foundation of web development is HTML, which enhances the visual display and interactive capabilities of webpages when combined with CSS and JavaScript. For web developers and designers, HTML is a vital tool because of its ease of use and broad support.
- **CSS:** A stylesheet language called CSS, or Cascading Style Sheets, is used to specify how an HTML or XML page is presented. It regulates a webpage's overall visual look, colour scheme, font choice, and layout. Developers may

construct aesthetically pleasing and consistent user interfaces across many devices and screen sizes by using CSS, which separates content from design. CSS is an essential technique in web development for generating adaptable and visually beautiful websites since it increases the flexibility and maintainability of online pages.

- **JavaScript:** High-level, flexible programming languages like JavaScript are frequently employed in web development to produce dynamic, interactive user interfaces. It enables the implementation of sophisticated features on webpages, including asynchronous content updates, form validations, and animations. JavaScript enhances the functionality of websites and makes it possible to create fully complete web apps by integrating with HTML and CSS with ease. JavaScript's extensive ecosystem, which includes frameworks and libraries like React, Angular, and Node.js, has made it indispensable for front-end and back-end development.
- **Bootstrap:** An open-source framework that is well-liked for creating mobile-first and responsive websites is called Bootstrap. It offers a selection of CSS and JavaScript elements and tools, like forms, modals, buttons, grids, and navigation bars, to make web page creation easier. Developers may assure uniformity and efficiency in their projects by adopting Bootstrap, which offers a predetermined collection of styles and components.
- **Flask:** Flask is a Python web framework that is lightweight and adaptable, making it simple to create online applications rapidly and with little overhead. Flask, which is well-known for its simplicity and flexibility, offers web development necessities like routing, request processing, and templating without imposing any specific project structure or dependencies. Because of this, it may be highly customised and used for a variety of purposes, from straightforward prototypes to intricate online services.
- **VS code:** Microsoft created Visual Studio Code (VS Code), a well-liked and adaptable code editor. It has capabilities like Git integration, intelligent code completion, syntax highlighting, and debugging. VS Code has a robust ecosystem of extensions that allow it to be tailored to a wide range of development requirements, including data science and web and application development.

Data Flow Diagram

Fig 3.1

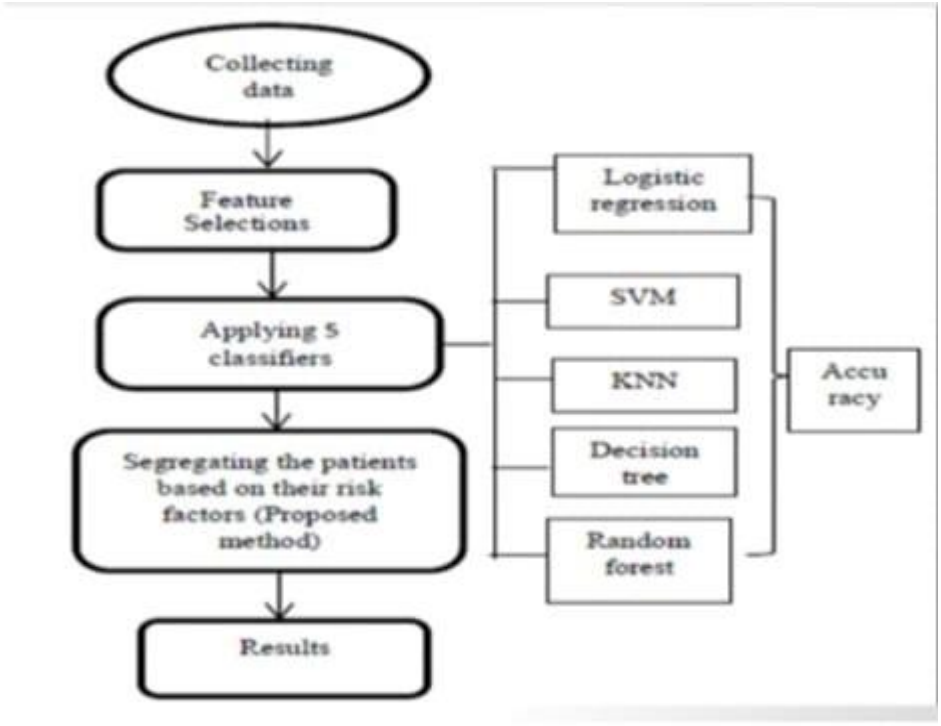

Fig 3.2

The first step in the process is gathering a dataset on heart disease. Next comes data preparation, which involves cleaning the data, dealing with missing values, and organizing both numerical and categorical information. To choose pertinent characteristics for the model, feature extraction is then carried out. Machine learning methods, such as Random

Forest (RF), K-Nearest Neighbors (KNN), Decision Tree (DT), Support Vector Machine (SVM), and Logistic Regression (LR), are trained and tested using the retrieved features. To forecast cardiac disease and measure performance, the models are implemented and assessed. Predicting heart disease and evaluating model performance are the last steps in figuring out how successful the models are.

Collecting Data: Gathering pertinent patient data is the first step in the procedure. Various health indicators, demographic data, medical history, and other relevant variables that may affect patient risk levels may be included in this data.

Feature Selection: Feature selection is the next action to be taken after gathering the data. This entails locating and picking the dataset's most important characteristics or factors that have the most bearing on estimating patient risk. Feature selection enhances the performance of machine learning models by lowering the dimensionality of the input.

Applying 5 Classifiers: To create prediction models, the chosen characteristics are subsequently input into five distinct machine learning classifiers. Among these classifiers are:

- **Logistic Regression:** A statistical method for predicting binary outcomes.
- **Support Vector Machine (SVM):** A classifier that finds the hyperplane that best separates the classes.
- **Decision Tree:** A model that uses a tree-like graph of decisions and their possible consequences.
- **K-Nearest Neighbour (KNN):** A non-parametric method used for classification by comparing the distance between data points.
- **Random Forest:** An ensemble learning method that constructs multiple decision trees and outputs the mode of the classes.

Accuracy Evaluation: The performance of each classifier is assessed according to how well it predicts the risk variables. Since accuracy establishes the dependability of the model's predictions, accuracy is an important parameter.

CHAPTER 4

METHADOLOGY

4.1 Overview

The objective of the heart disease prediction project is to develop a strong machine learning model that can forecast the risk of heart disease based on 13 essential health characteristics, such as age, blood pressure, and cholesterol levels. The first step of the project is gathering and preparing data to fill in any missing values and standardise feature ranges. Next, feature engineering is used to identify the most pertinent properties and produce any additional features that might enhance model performance. Performance measures like as accuracy, precision, and recall are used in the training and evaluation of a variety of machine learning methods, including Gradient Boosting, Random Forests, and Logistic Regression. To make sure it adapts effectively to fresh, untested data, the model is put through a thorough testing process on a different dataset. After the best model has been determined, It is integrated into an actual application, enabling continuous performance monitoring and forecasts in real time.

4.2 Datasets

The heart disease dataset is an extensive set of patient information that is utilized to forecast the occurrence of heart disease.

1. **Age:** Age is a significant risk factor for heart disease; as age increases, the risk of developing heart disease generally increases.
2. **Sex:** Sex can influence the prevalence and presentation of heart disease. Men often show symptoms earlier than women, but women may have different risk factors and symptoms
    - 1 = Male and 0 = Female
3. **Chest Pain Type:** Different types of chest pain are associated with varying risks of heart disease. Typical angina is often a stronger indicator of heart disease.
    - 0: Atypical Angina
    - 1: Non-Anginal Pain
    - 2: Typical Angina

- 3: Asymptomatic Angina

4. **Serum Cholesterol:** High serum cholesterol is a major risk factor for heart disease as it can lead to the buildup of plaque in arteries.
5. **Resting Blood Pressure:** High resting blood pressure is a significant risk factor for cardiovascular disease.
6. **Fasting Blood Pressure:** Elevated fasting blood sugar levels are indicative of diabetes, which increases the risk of heart disease.
    - 1: True (fasting blood sugar ≥ 120 mg/dL)
    - 0: False (fasting blood sugar < 120 mg/dL)
7. **Resting Electrocardiogram (ECG):** Abnormal ECG results can indicate heart conditions such as left ventricular hypertrophy or ischemia.
    - 0: Normal
    - 1: ST-T Wave Abnormality
    - 2: Left Ventricular Hypertrophy
8. **Maximum Heartrate:** A higher heart rate can indicate better fitness, but very high heart rates or abnormal responses to exercise can signal heart issues.
9. **Exercise-Induced Angina:** Exercise-induced angina is a common symptom of coronary artery disease.
    - 1: Yes
    - 0: No
10. **Slope of Peak Exercise ST Segment:** The slope of the ST segment can help identify abnormalities that are indicative of ischemia.
    - 1: Upsloping
    - 2: Flat
    - 3: Down sloping
11. **ST Depression Induced by Exercise:** ST segment depression is a sign of ischemia or heart disease.
12. **Number of Major Vessels Coloured by Fluoroscopy:** The number of coloured vessels can indicate the extent of coronary artery disease. Fewer visible vessels suggest greater severity.
    - 0 to 3 No of vessels
13. **Thalassemia Status:** Thalassemia can affect haemoglobin and is considered in evaluating overall cardiovascular health.

- 1: Reversible Defects
- 2: Permanent Defects
- 3: Normal

14. **Target Variable (Presence of Cardiac Disease):** This is the outcome variable that your model is trying to predict based on the other features.
    - 1: Presence of Disease
    - 0: Absence of Disease

It is available in several repositories, including **Kaggle** and the **UCI Machine Learning Repository**, where academics and data scientists may test algorithms and increase predicted accuracy with its help.

No	Attributes	Description
1	Age	Age in year
2	Sex	0 for female and 1 for male
3	Cp	Chest pain type Value 1: typical angina Value 2: atypical angina Value 3: non-anginal pain Value 4: asymptomatic
4	Trestbps	Resting blood sugar in mm Hg on admission to the hospital
5	Chol	Serum cholesterol in mg/dl
6	Fbd	(Fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
7	Restecg	Resting ECG result
8	Thalach	Maximum heart rate achieved
9	Exang	Exercise induced angina
10	Oldpeak	ST depression induced by exercise relative to rest
11	Slope	Slope or peak exercise ST segment
12	Ca	Number of major vessels colored by fluoroscopy
13	Thal	Defect type
14	num	The predicted attribute

Fig 4.1

4.3 Data Information and Visualisation

A wide range of characteristics that affect cardiovascular health are included in the dataset, which primarily focuses on heart disease prediction. Important characteristics include age and sex information, clinical measurements (blood pressure, cholesterol, fasting blood sugar), and test results from diagnostic tests (heart rate, electrocardiogram, exercise-induced angina, exercise ST segment slope, ST depression, number of major vessels coloured by fluoroscopy, and thalassemia status). While numerical factors like age and cholesterol levels give quantitative information, categorical variables like sex and the kind of chest discomfort provide qualitative data.

The distribution of numerical characteristics in the dataset may be shown using histograms, outliers can be found using box plots, and variable relationships can be seen with scatter plots. Furthermore, relationships between features may be displayed using heatmaps, and the frequency distribution of categorical variables can be shown using bar charts.

Sl.no	Information Type	values
1	Number Data rows	2623
2	Number of Data columns	14
3	Number of Attributes	13
4	Number of Disease present values	1337
5	Number of Disease absent values	1286

Table 4.1 Basic Information

For exploratory data analysis, the df. describe () function is essential since it provides a succinct summary of the numerical properties of the dataset. It aids in evaluating data distributions, identifying anomalies, and comprehending the range of values by summarizing statistical variables like mean, median, standard deviation, and quartiles. This synopsis supports the identification of patterns, possible problems with data quality, and guides choices about preprocessing operations such as normalization or scaling. All things considered, it is a fundamental tool for understanding the data and directing more research or model building.

STATSTICAL INFORMATION OF DATA SET								
ATTRIBUTES OF DISEASE	**count**	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**
AGE	2623	54.42623	9.072037	29	48	56	61	77
SEX	2623	0.692337	0.461614	0	0	1	1	1
CHEST PAIN	2623	1.17499	1.225629	0	0	1	2	4
REST BLOOD PRESSURE	2623	131.5856	17.54507	94	120	130	140	200
SERUM CHOLESTEROL	2623	246.4072	51.61195	126	211	240	275	564
FASTING BLOOD SUGAR	2623	0.149066	0.356221	0	0	0	0	1
REST ELECTROCARDIOGRAPH	2623	0.580252	0.611627	0	0	1	1	2
MAX HEART RATE	2623	149.2337	22.99859	71	132	152	166	202
EXERCISE-INDUCED ANGINA	2623	0.334731	0.471986	0	0	0	1	1
ST DEPRESSION	2623	1.065612	1.169802	0	0	0.8	1.8	6.2
SLOP	2623	1.407549	0.619826	0	1	1	2	3
NO. OF VESSELS	2623	0.742661	1.021003	0	0	0	1	4
THALASSEMIA	2623	2.566908	1.118462	0	2	2	3	7
TARGET	2623	0.509722	0.500001	0	0	1	1	1

Table 4.2 Describing the dataset

4.3.1 Data Visualization

Numerical and categorical data are transformed into visual representations like bar charts, scatter plots, histograms, and heatmaps through data visualization. Finding patterns, correlations, and outliers—which might not be immediately obvious from raw data alone—is made easier with the help of these visual tools. Data visualization improves comprehension and interpretation by enabling us to swiftly assimilate vast amounts of information.

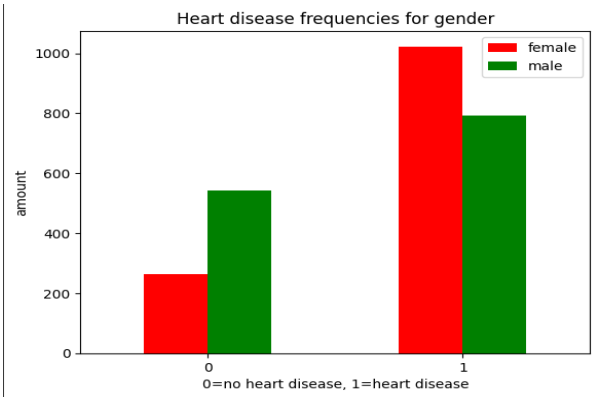

Fig 4.2

The comparison between the male and female genders about the target variable will be shown visually in the previously described picture.

Corelation Matrix: A table that shows the correlation coefficients between several variables in a dataset is called a correlation matrix. The matrix's individual cells, which normally range from -1 to 1, indicate the direction and intensity of the linear connection between two variables. Strong positive correlations are shown by coefficients near 1, which means that as one variable rises, the other also tends to rise. On the other hand, a significant negative correlation is shown by a coefficient close to -1, meaning that one variable grows while the other declines. A number around 0 denotes the absence of a linear connection. In statistical modelling or machine learning, correlation matrices can help with feature selection, multicollinearity, and possible interaction decisions by showing which variables are connected to one another.

Heat Map: A heat map is a type of data visualization where values are represented in a matrix or grid style using colour. The colour of each cell in the heat map corresponds to the value it represents; a gradient of colours denotes various data ranges. When presenting complicated data sets, this representation is very helpful for immediately identifying trends, connections,oranomalies.

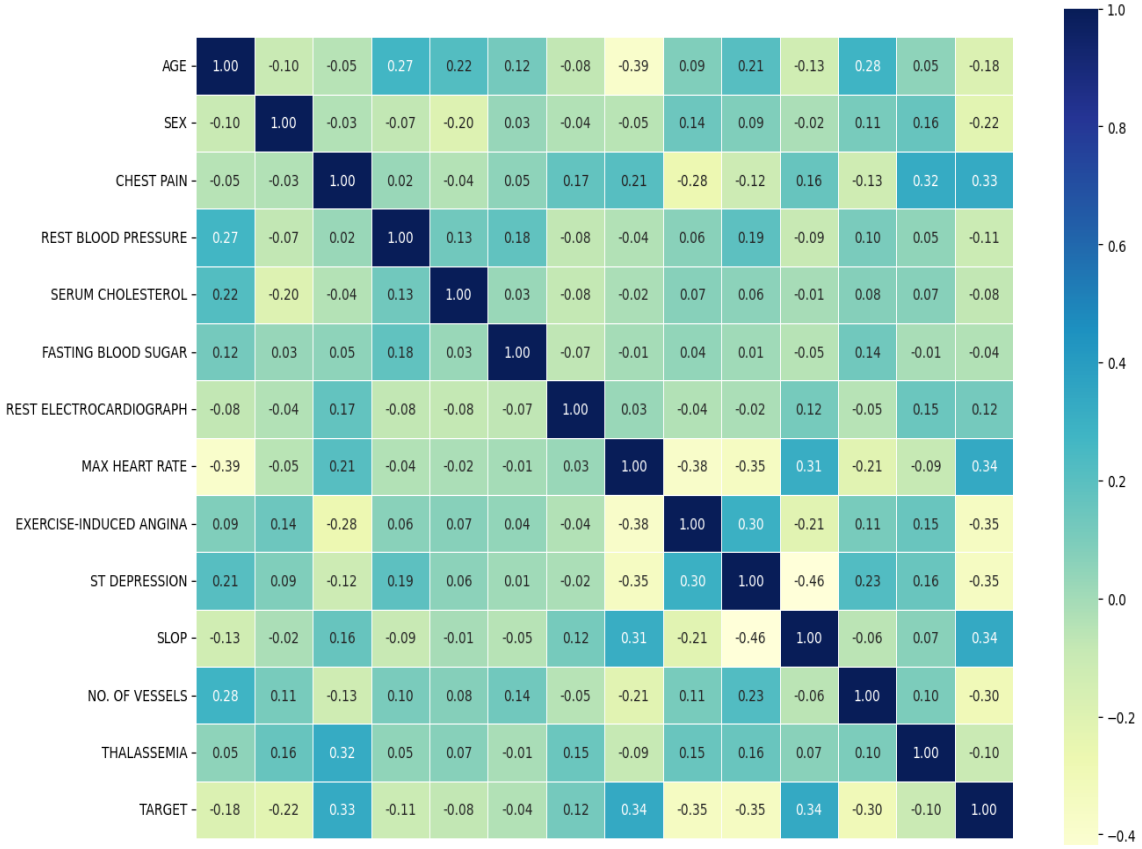

Fig 4.3

4.3 Data Preprocessing Techniques

An essential first step in getting data ready for machine learning models is data preparation. It entails converting unstructured data into a clear, interpretable format, frequently involving the management of missing values, the encoding of categorical categories, the scaling of numerical characteristics, and other tasks. Ensuring that the characteristics are in a uniform format and size is the aim in preparing the data for modelling.

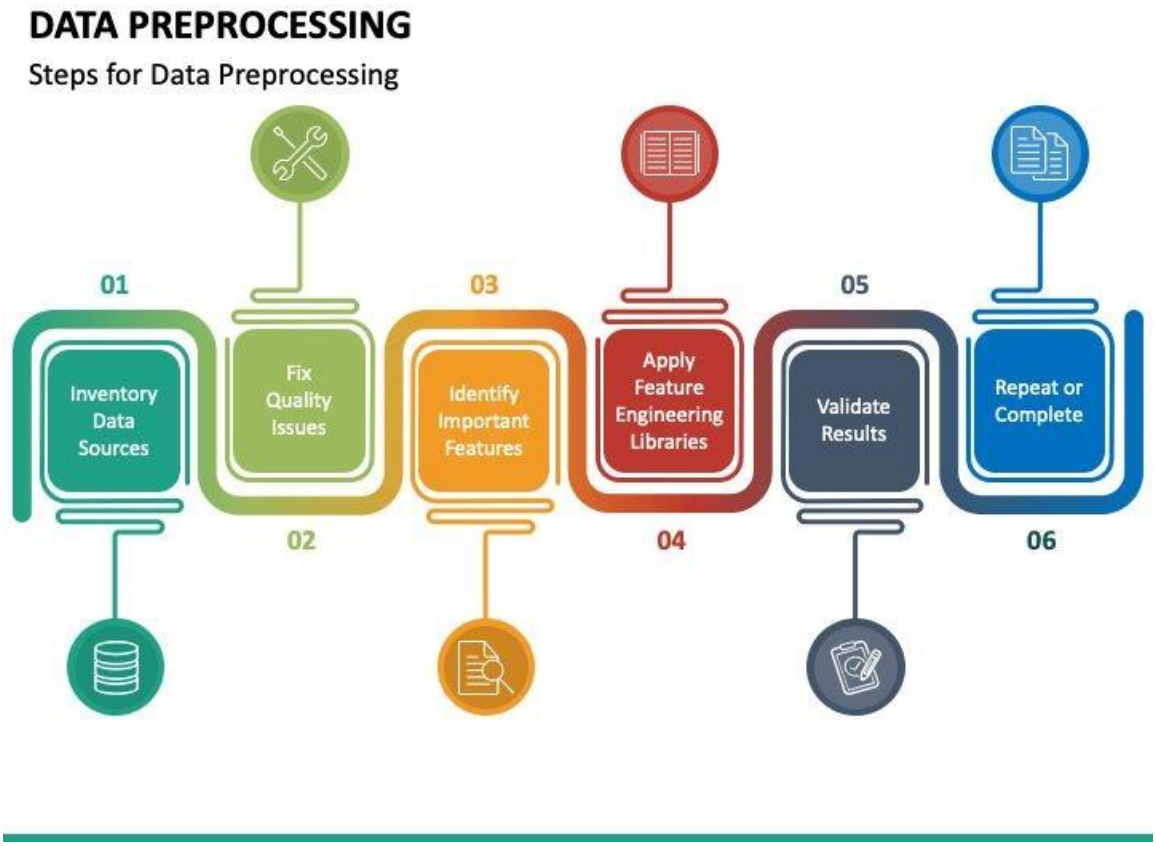

Fig 4.4

Numbered and Categorical Columns

categorical_cols: The category columns in the dataset are listed in this list. Discrete and sometimes non-numeric categorical variables represent labels or categories. **numerical_cols:** This list contains numerical columns. Numerical variables are inherently quantitative and can be either discrete or continuous. Often, features need to be scaled to ensure that each one contributes equally to the learning process of the model.

Numerical Data Preprocessing:

- **Standard Scaling (StandardScaler):** This technique standardizes features by removing the mean and scaling to unit variance. The formula used is (X - mean) / standard deviation. This is essential for algorithms that assume normally distributed data (e.g., linear regression) or those sensitive to the scale of data (e.g., neural networks).

Categorical Data Preprocessing:

- **One-Hot Encoding (OneHotEncoder):** This converts categorical variables into a binary (0 or 1) format. For each unique category, a new binary column is created. This encoding is particularly useful for models that cannot directly handle categorical data. The handle unknown='ignore' parameter ensures that unseen categories during training do not cause errors during transformation.

Pipeline Concept: A pipeline in machine learning is a sequence of data processing steps arranged to automate the workflow. It ensures that each step is executed in the correct order and that the transformations applied to the training data are also applied to any new data.

4.4 Model Selection

Model selection refers to the process of choosing the most appropriate machine learning model for a given problem. This involves evaluating various models and selecting the one that best fits the data and performs well on the task at hand. In the context of a pipeline, model selection is integrated into the workflow, allowing for streamlined training, evaluation, and hyperparameter tuning.

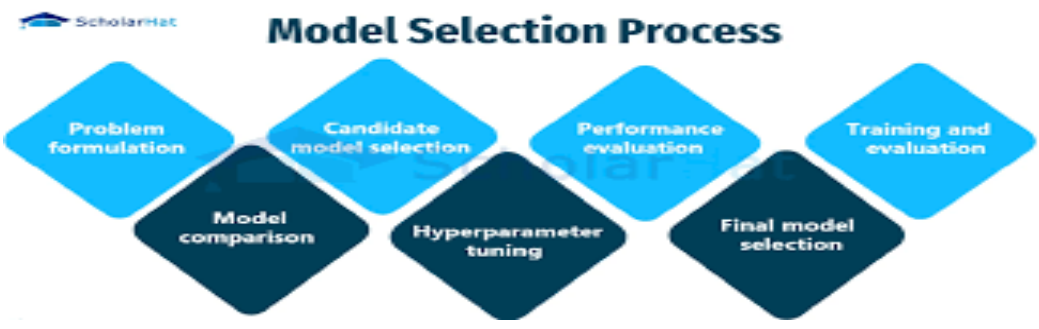

Fig 4.5

Logistic Regression: An approach used in statistics and machine learning for binary classification tasks-tasks where the result falls into one of two potential classes—is called logistic regression. Logistic regression uses a logistic function (also called the sigmoid function) to predict the likelihood of a categorical result as opposed to linear regression, which predicts a continuous output. Any real-valued integer can be mapped by this function to the range [0, 1], which represents the default class probability. The model is especially helpful for cases when the response variable is categorical since it maximizes the chance of witnessing the provided data to estimate the parameters.

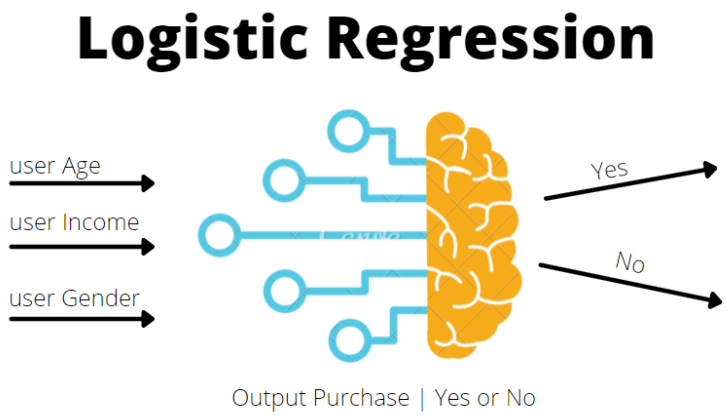

Fig 4.6

Decision Tree: One machine learning technique that is utilized for both regression and classification applications is the decision tree. In order to create a decision tree model, it divides the dataset into subgroups according to the values of the input characteristics. A feature is represented by an internal node, a decision rule by a branch, and a result or class label by a leaf node.

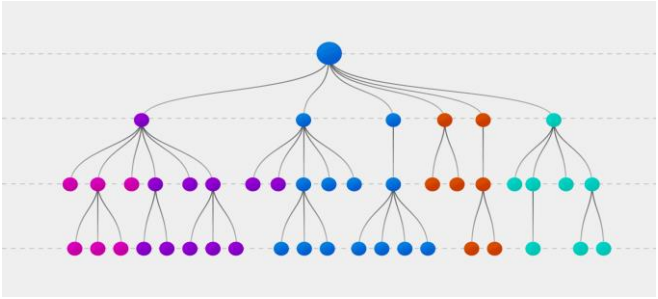

Fig 4.7

Random Forest Classifier: An ensemble learning method used for classification problems is called a Random Forest classifier. In order to function, it builds a large number of decision trees during training and outputs the mean prediction for regression or the mode of the classes for classification based on each individual tree. The term "random" refers to two aspects of the algorithm: first, it chooses a random subset of features for every split in a tree; second, it bootstraps, or utilizes a random sampling of data points (with replacement) to train each tree.

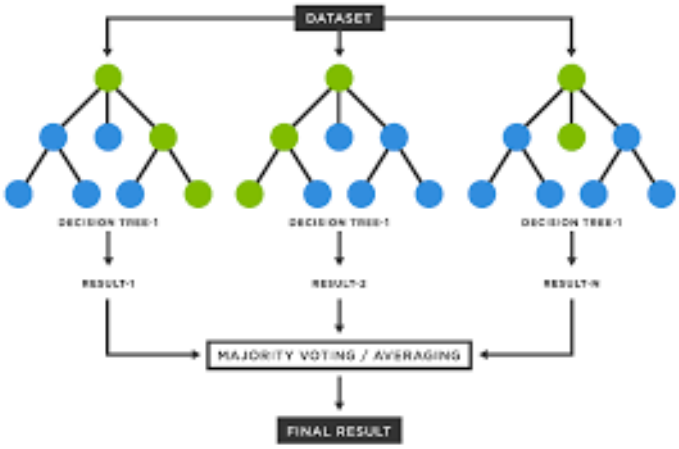

Fig 4.8

K-Nearest Neighbours: A straightforward, instance-based, non-parametric learning approach for classification and regression problems is the k-Nearest Neighbours (k-NN) classifier. When it comes to categorization, the algorithm groups data points according to the classifications of their neighbours. It determines the distance typically using metrics like Euclidean distance between the data point and every other point in the training set. The number of nearest neighbours to consider while classifying a given set of data is indicated by the letter "k" in k-Nowhen a new data point needs to be classified.

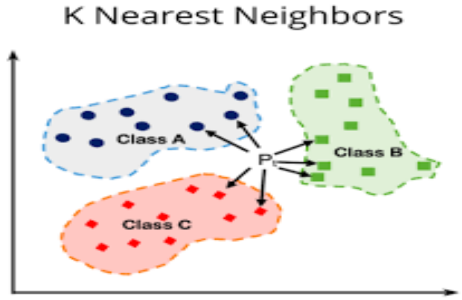

Fig 4.9

Support Vector Classifiers: For classification problems, the supervised machine learning algorithm Support Vector Classifier (SVC) is employed. Its foundation is the idea of determining which hyperplane in a feature space optimally divides the data points of several classes. The distance between the nearest data points (support vectors) of various classes and the hyperplane is called the margin, and it is the margin that the SVC seeks to optimize. Better generalization on unknown data may be achieved with the aid of this maximizing. Using kernel functions, which convert the input data into a higher-dimensional space where a linear separator may be located, SVC can also handle non-linear classification issues.

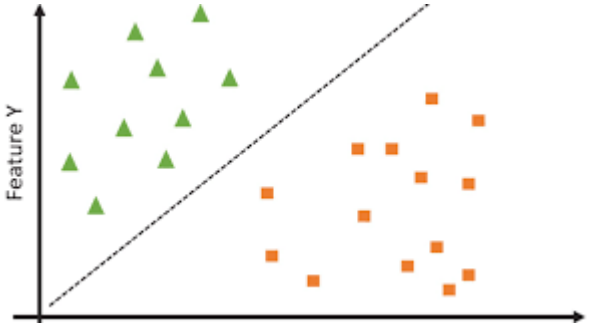

Fig 4.10

Gaussian Navie Bayes: Based on Bayes' theorem, the probabilistic classification method Gaussian Naive Bayes (Gaussian NB) is best suited for scenarios in which the features are continuous and presumed to have a Gaussian (normal) distribution. By assuming that characteristics are conditionally independent given the class name, it lowers the modeling complexity and streamlines the classification process. With Gaussian NB, the training data is used to estimate the mean and variance of a Gaussian distribution, which represents the probability of the features. Based on the observed feature values, the algorithm calculates the posterior probability of each class.

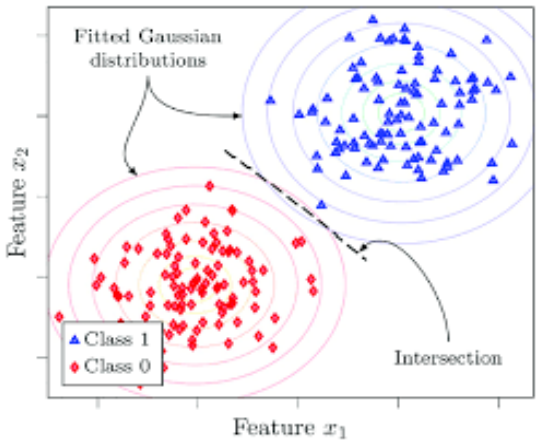

Fig 4.11

4.5 Justice for Model Selection

Robust and adaptable, the Random Forest Classifier is an excellent machine learning technique for handling tasks involving both regression and classification. In order to increase accuracy and reduce overfitting, it builds many decision trees during training and aggregates their predictions. A random subset of the data and characteristics are used to build each tree, adding variation, and improving the model's capacity for generalization. **"The ensemble of trees majority vote determines the ultimate categorization. With this method, Random Forest can efficiently handle big datasets with lots of dimensions and intricate feature interactions."**

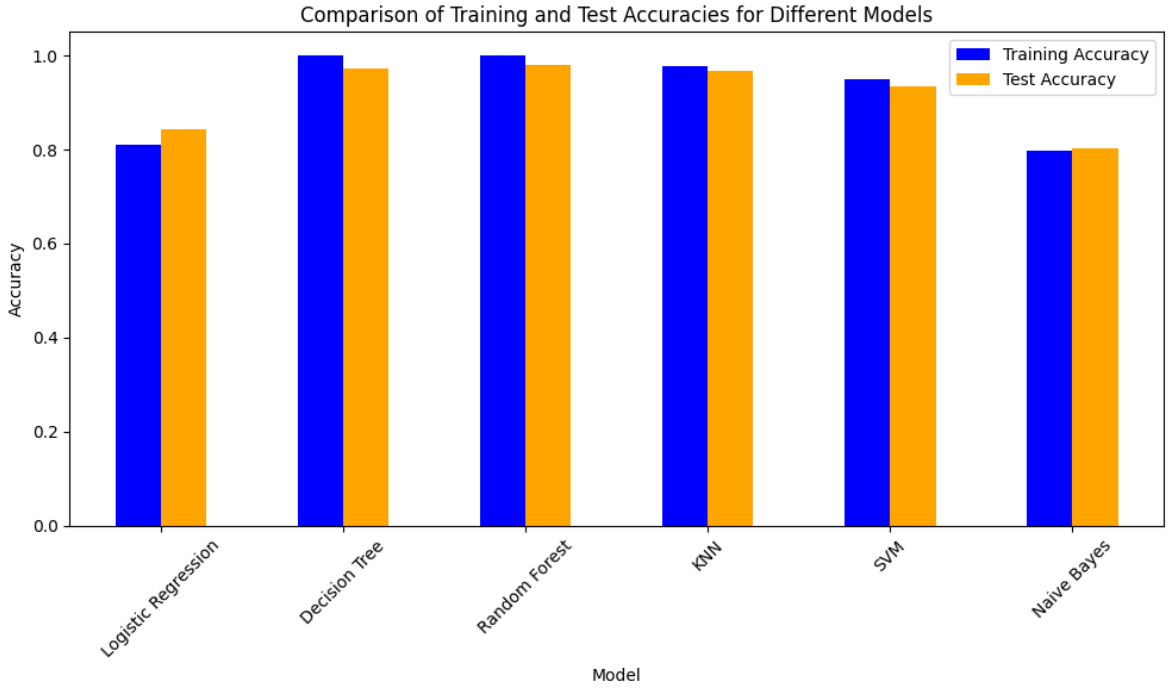

Fig 4.12

Logistic Regression

- Trained_data_accuaracy:81%
- Test_data_accuaracy:84%

K-Nearest Neighbour

- Trained_data_accuaracy:97.77%
- Test_data_accuaracy:97.16%

Support Vector Classifier

- Trained_data_accuaracy:95.12%
- Test_data_accuaracy:93.56%

Navie basin

- Trained_data_accuaracy:97.21%
- Test_data_accuaracy:80.76%

Decision Tree

- Trained_data_accuaracy:100%
- Test_data_accuaracy:97.33%

Random Forest

- Trained_data_accuaracy:100%
- Test_data_accuaracy:97.77%

Based on the accuracy of all six machine algorithms—logistic regression, Random Forest, SVC, Decision Tree, K-Nearest Neighbour, and Navie Bayes the **RANDOM FOREST** algorithm exhibits the maximum accuracy to our dataset. Thus, we decide to use the random forest model to forecast my cordial infractions.

4.6 Model Architecture

An ensemble of decision trees, each constructed independently from random subsets of the data and characteristics, make up the architecture of a Random Forest model. Bootstrap sampling is used to build numerous decision trees during training, with each tree being trained on a distinct random sample of the training set with replacement. To offer even more variation, each node in the trees bases its judgments on a randomly selected subset of characteristics. After training, the model averages (or votes in the case of classification tasks) or aggregates the predictions from all the trees. When compared to individual decision trees, this ensemble technique increases resilience, decreases overfitting, and improves accuracy.

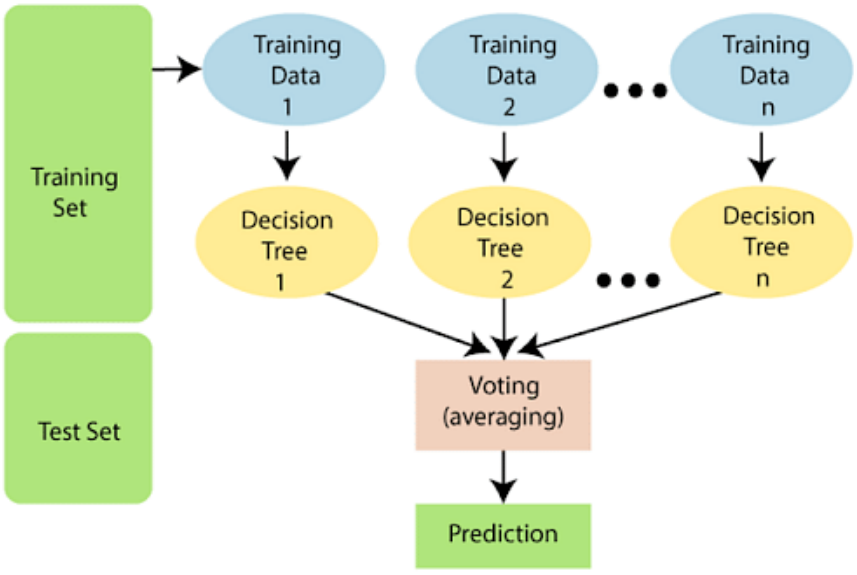

Fig4.13

A potent tree learning method in machine learning is the Random Forest algorithm. During the training stage, it generates many Decision Trees. To measure a random subset of

characteristics in each partition, a random subset of the data set is used to build each tree. Because each tree is more variable because of the randomization, there is less chance of overfitting and overall prediction performance is enhanced.

Algorithm for Random Forest Work:

Step 1: Select random K data points from the training set.

Step 2: Build the decision trees associated with the selected data points(Subsets).

Step 3: Choose the number N for decision trees that you want to build.

Step 4: Repeat Step 1 and 2.

Step 5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Key Features of Random Forest

- **High Predictive Accuracy:** Imagine Random Forest as a team of decision-making wizards. Each wizard (decision tree) looks at a part of the problem, and together.
- **Resistance to Overfitting:** Random Forest is like a cool-headed mentor guiding its apprentices (decision trees). Instead of letting each apprentice memorize every detail of their training, it encourages a more well-rounded understanding.
- **Large Datasets Handling:** Dealing with a mountain of data? Random Forest tackles it like a seasoned explorer with a team of helpers (decision trees).
- **Variable Importance Assessment:** Think of Random Forest as a detective at a crime scene, figuring out which clues (features) matter the most.
- **Built-in Cross-Validation:** Random Forest is like having a personal coach that keeps you in check. As it trains each decision tree, it also sets aside a secret group of cases (out-of-bag) for testing.
- **Handling Missing Values:** Life is full of uncertainties, just like datasets with missing values. Random Forest is the friend who adapts to the situation, making predictions using the information available.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 Model Deployment

The act of incorporating a machine learning model that has been trained into a production setting so that it may make predictions on fresh data is known as model deployment. This entails moving the model from a development environment into a live environment while making sure it operates effectively and consistently. Important actions include deciding on the right infrastructure, configuring interfaces or APIs, keeping an eye on the model's performance, and updating it to make sure it stays correct over time. Real-time decision-making and the use of model-derived insights to address real-world issues are made possible by successful deployment.

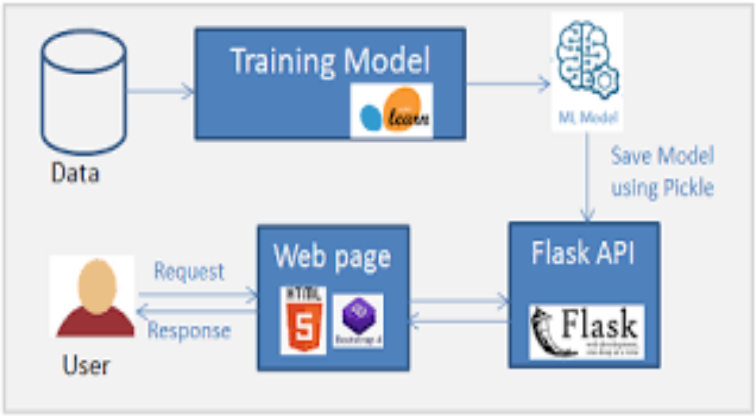

Fig 5.1

Using Flask for model deployment entails creating a Flask web application to provide your machine learning model that has been trained. Developing HTML files for the user interface and Python scripts to load the model and manage prediction requests are also steps in the process. User input is routed from HTML forms to the model by the Flask application, which then evaluates the inputs and provides the user with predictions. This configuration makes it simple to interact with the model via a web browser, enabling integration with other applications and real-time use.

Model.py

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
import pickle
import csv
def train_and_save_model():
    data = pd.read_csv(r"D:\miniproject\miniproject (3)\miniproject
(2)\Fmini\miniproject (3)\heart.csv")
    X = data.drop('TARGET', axis=1)
    y = data['TARGET']
    X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=109)
    clf = RandomForestClassifier(n_estimators=100, random_state=42)
    clf.fit(X_train, y_train)
    with open('random_forest_model.pkl', 'wb') as model_file:
        pickle.dump(clf, model_file)
def random_forest_pred(age, sex, cp, trestbps, chol, fbs, restecg, thalach,
exang, oldpeak, slope, ca, thal):
    with open('random_forest_model.pkl', 'rb') as model_file:
        clf = pickle.load(model_file)
    new_data = [age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang,
oldpeak, slope, ca, thal]
    result = clf.predict([new_data])[0]
    with open("heart.csv", "a", newline='') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow([age, sex, cp, trestbps, chol, fbs, restecg,
thalach, exang, oldpeak, slope, ca, thal, result])
    return result
train_and_save_model()
prediction = random_forest_pred(age, sex, cp, trestbps, chol, fbs, restecg,
thalach, exang, oldpeak, slope, ca, thal)
print("Prediction:", prediction)
```

With Pickle, you can save machine learning models to a file and load them later for predictions without having to retrain them. Pickle is a Python package used for serializing and deserializing objects. The pickle is used in the procedure. Pickle. Load() reads the byte stream from the file and recreates the model, whereas dump () transforms the trained model into a byte stream and saves it to a file. The model may be easily stored, shared, and deployed across many contexts and applications thanks to its serialized form, which guarantees that all the model's parameters and learned patterns are maintained.

Main.py

```
from flask import Flask, render_template, request, jsonify
import os
import numpy as np
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
import pickle
app = Flask(__name__)
speech, name, age1, gender = "", "", "", ""
problem, sym_duration = "", ""
q1, q2, q3, q4, q5, q6 = 0, 0, 0, 0, 0, 0
sym1, sym2, sym3 = "", "", ""
new_report, tips, soln = "", "", ""
doctor, date, time = "", "", ""
cp_mapping = {
    'Typical Angina': 0,
    'Atypical Angina': 1,
    'Non-anginal Pain': 2,
    'Asymptomatic': 3
}
restecg_mapping = {
    'Normal': 0,
    'Having ST-T wave abnormality': 1,
    'Left ventricular hypertrophy': 2
}
slope_mapping = {
    'Upsloping': 0,
    'Flat': 1,
    'Downsloping': 2
}
ca_mapping = {
    '0': 0,
    '1': 1,
    '2': 2,
    '3': 3,
    '4': 4
}
thal_mapping = {
    'Normal': 1,
    'Fixed Defect': 2,
    'Reversible Defect': 3
}
yes_no_mapping = {
    'Yes': 1,
    'No': 0
}
@app.route('/home')
def Home():
    return render_template('home.html')
@app.route('/about')
def pass_page():
    return render_template('about.html')
@app.route('/aboutus')
def aboutus():
    return render_template('aboutus.html')
```

```
@app.route('/form')
def form():
    return render_template('form.html')
@app.route('/', methods=['GET', 'POST'])
def getvalue():
    if request.method == 'POST':
        try:
            age1 = int(request.form.get('age'))
            gender = request.form.get('sex')
            cp = cp_mapping.get(request.form.get('cp'))
            trestbps = float(request.form.get('trestbps'))
            chol = float(request.form.get('chol'))
            fbs = yes_no_mapping.get(request.form.get('fbs'))
            restecg = restecg_mapping.get(request.form.get('restecg'))
            thalach = float(request.form.get('thalach'))
            exang = yes_no_mapping.get(request.form.get('exang'))
            oldpeak = float(request.form.get('oldpeak'))
            slope = slope_mapping.get(request.form.get('slope'))
            ca = ca_mapping.get(request.form.get('ca'))
            thal = thal_mapping.get(request.form.get('thal'))
            gender_numeric = 1 if gender.lower() == 'male' else 0
            try:
                op = random_forest_pred(age1, gender_numeric, cp,
trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal)
                print(f"Prediction: {op}")
            except Exception as e:
                print(f"Prediction error: {e}")
                return "Error occurred during prediction: " + str(e)
            if op == 0:
                opstr = "No Heart Disease"
                speech = "Your report looks fine."
            elif op == 1:
                opstr = "Heart Disease Present"
                speech = "You may be suffering from a heart
disease/problem!"
            return render_template('form.html', prediction=opstr,
name=request.form.get('name'), age=age1, sex=gender,
                                   cp=request.form.get('cp'),
trestbps=trestbps, chol=chol, fbs=request.form.get('fbs'),
                                   restecg=request.form.get('restecg'),
thalach=thalach, exang=request.form.get('exang'),
                                   oldpeak=oldpeak,
slope=request.form.get('slope'), ca=request.form.get('ca'),
                                   thal=request.form.get('thal'))
        except ValueError as e:
            print(f"ValueError: {e}")
            return "Invalid input values. Please check your input."
    else:
        return render_template('home.html')
@app.route('/webhook', methods=['POST'])
def webhook():
    req = request.get_json(silent=True, force=True)
    res = makeWebhookResult(req)
    res = jsonify(res)
    return res
def makeWebhookResult(req):
```

```
    query_response = req.get("queryResult", {})
    fulfillment_text = ""
    global name, gender, age1, sym1, sym2, new_report
    if query_response.get("action") == "user_name":
        name = query_response.get("parameters", {}).get("given-name")
    if query_response.get("action") == "user_age":
        age1 = int(query_response.get("parameters", {}).get("age",
{}).get("amount", 0))
    if query_response.get("action") ==
"DefaultWelcomeIntent.DefaultWelcomeIntent-custom.Checkup_Patient-
custom":
        gender = query_response.get("parameters", {}).get("Gender")
        fulfillment_text = f"OK {name} ({age1}), please fill and submit
the report form on the right side."
    return {"fulfillmentText": fulfillment_text}
def random_forest_pred(age, sex, cp, trestbps, chol, fbs, restecg,
thalach, exang, oldpeak, slope, ca, thal):
    with open('random_forest_model.pkl', 'rb') as model_file:
        clf = pickle.load(model_file)
    input_data = np.array([[age, sex, cp, trestbps, chol, fbs, restecg,
thalach, exang, oldpeak, slope, ca, thal]])
    prediction = clf.predict(input_data)
    return prediction[0]
if __name__ == '__main__':
    port = int(os.getenv('PORT', 5000))
    app.run(debug=True, port=port, host='0.0.0.0')
```

Using a Random Forest classifier, this Flask application functions as a web interface for heart disease prediction. It has many route handlers (home, about, aboutus, and form) for various web pages. User input is gathered on the form page and processed by the getvalue function upon form submission. Numerical values appropriate for the machine learning model are mapped from user inputs. Using user-supplied data, the random forest pred function imports a pre-trained Random Forest model from a pickle file (random forest model.pkl) and employs it to forecast the existence of heart disease. The user is presented with a presentation of the prediction findings that shows whether cardiac disease is present**.** In order to communicate with a conversational agent, the application also has a webhook handler. This feature enables the collection of user input and the generation of answers depending on predetermined actions. The Flask application can be accessed with a web browser since it runs on a designated port.

5.2 Screenshots

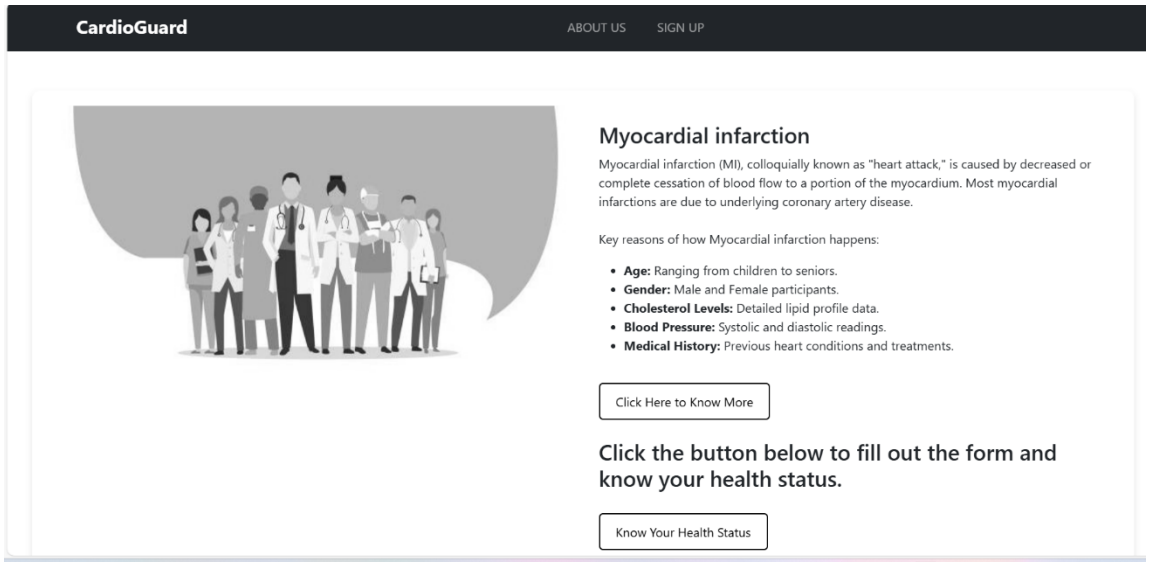

Fig 5.1

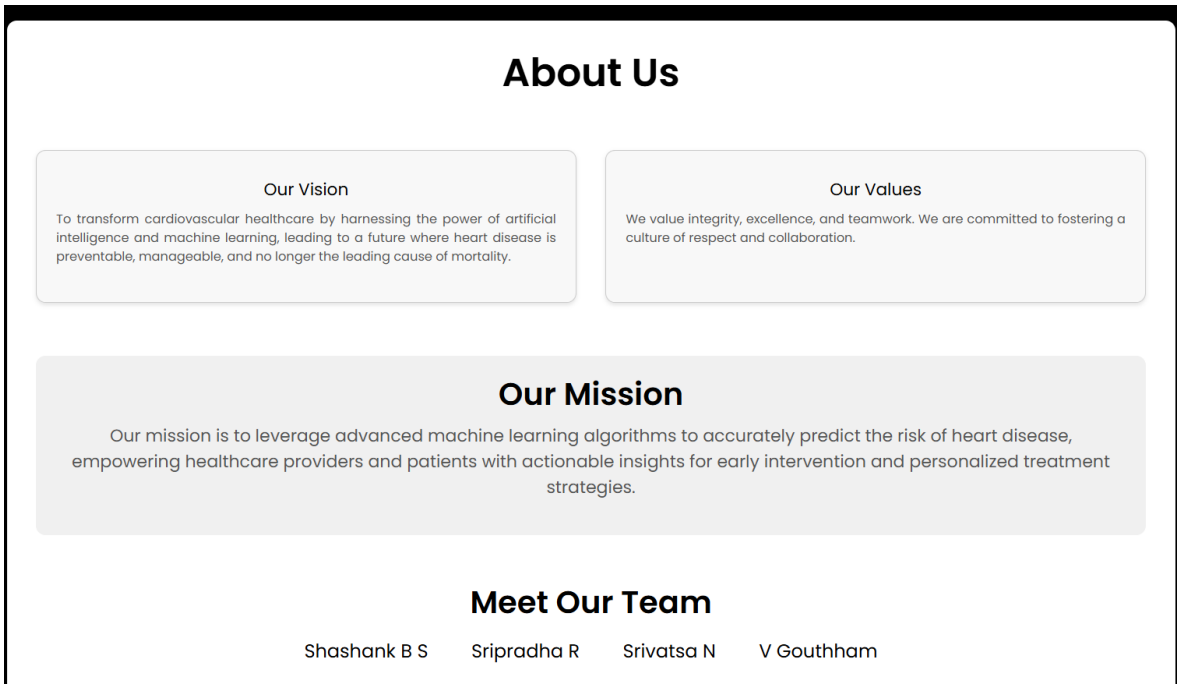

Fig 5.2

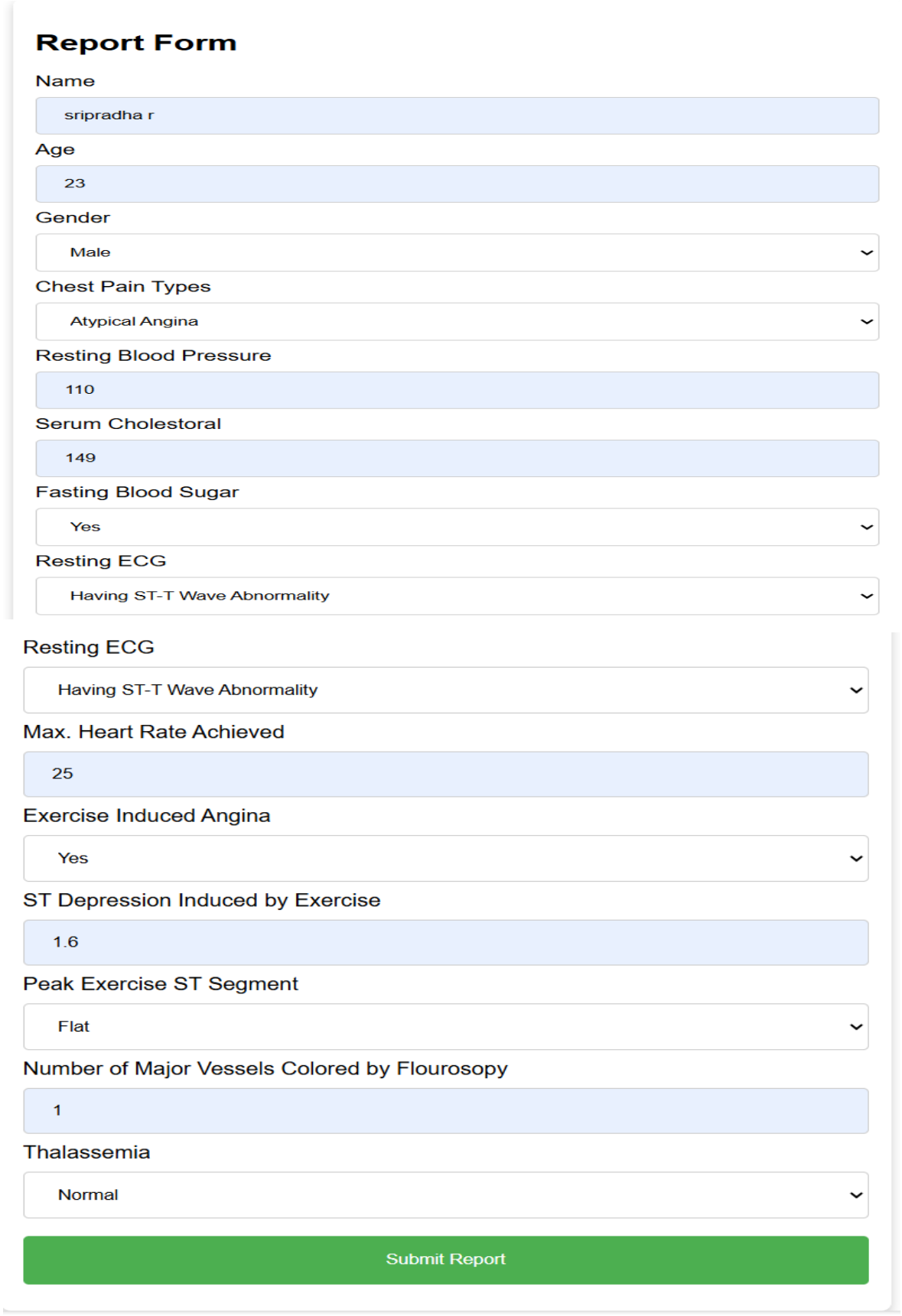
Report Form
Name
sripradha r
Age
23
Gender
Male
Chest Pain Types
Atypical Angina
Resting Blood Pressure
110
Serum Cholestoral
149
Fasting Blood Sugar
Yes
Resting ECG
Having ST-T Wave Abnormality
Resting ECG
Having ST-T Wave Abnormality
Max. Heart Rate Achieved
25
Exercise Induced Angina
Yes
ST Depression Induced by Exercise
1.6
Peak Exercise ST Segment
Flat
Number of Major Vessels Colored by Flourosopy
1
Thalassemia
Normal
Submit Report

Fig 5.3

OUTPUT

sripradha r, No Heart Disease

Your Submitted Details:

Field	Value
Name	sripradha r
Age	23
Gender	Male
Chest Pain Types	Atypical Angina
Resting Blood Pressure	110.0
Serum Cholestoral	149.0
Fasting Blood Sugar	Yes
Resting ECG	Having ST-T Wave Abnormality
Max. Heart Rate Achieved	25.0
Exercise Induced Angina	Yes
ST Depression Induced by Exercise	1.6
Peak Exercise ST Segment	Flat
Number of Major Vessels Colored by Flourosopy	1
Thalassemia	Normal

Fig 5.4

CONCLUSION

In summary, the Heart illness Prediction project effectively utilized machine learning methodologies to create a model that facilitates the early identification of heart illness. High accuracy and dependability were shown by the model, which made use of important characteristics including age, maximal heart rate, resting blood pressure, and cholesterol levels. The use of an intuitive front-end interface facilitates the input of patient data by healthcare professionals and provides them with instantaneous risk projections, therefore supporting prompt and potentially life-saving treatments. This initiative demonstrates the critical role that data-driven insights play in medical diagnostics and lays the groundwork for future improvements that will include larger datasets, more advanced algorithms, and seamless interaction with electronic health records. Overall, our research highlights how machine learning may greatly enhance healthcare outcomes by facilitating early detection accurate disease prediction

FUTURE ENHANCEMENT

- While our Heart Disease Prediction project has achieved significant milestones, there are several opportunities for future enhancement:
- Advanced Machine Learning Techniques: Incorporating more advanced algorithms, such as deep learning and ensemble methods, could improve the model's predictive accuracy and robustness.
- Larger and Diverse Datasets: Expanding the dataset to include more diverse patient populations and additional features, such as genetic information and lifestyle factors, can enhance the model's generalizability and accuracy.
- Integration with Electronic Health Records (EHR): Developing seamless integration with EHR systems will ensure real-time access to patient data, making the prediction model more practical and useful in clinical settings.
- Real-time Monitoring and Continuous Learning: Implementing real-time data monitoring and continuous learning mechanisms will allow the model to update and improve its predictions as new data becomes available.
- User Experience Improvements: Enhancing the user interface for better usability and accessibility can ensure that the tool is more widely adopted by healthcare

professionals.

- Validation and Testing: Conducting extensive validation and testing in real-world clinical environments can help fine-tune the model and ensure its reliability and effectiveness.
- Personalized Predictions: Incorporating personalized medicine approaches to tailor predictions based on individual patient profiles and history could further improve prediction accuracy and patient outcomes.
- By focusing on these enhancements, we can significantly advance the capability and impact of our heart disease prediction model, ultimately contributing to better healthcare delivery and patient care.

REFERENCES

- **https://github.com/Bakar31/Heart-Disease/blob/master/data/heart.csv**
- **https://github.com/amberkakkar01/Health-Care-Chatbot**
- **https://github.com/VenkateshBH99/Heart-and-Kidney-disease-prediction-Django/blob/master/predict_risk/machine_learning_models/svc.py**
- **https://iopscience.iop.org/article/10.1088/1757-899X/1022/1/012046**
- **https://f1000research.com/articles/11-1126**