HEART DISEASE PREDICTION
IMPROVED BY
RANDOM FOREST ALGORITHM

MINI PROJECT REPORT

Submitted by

BALAJEEY R K	**(721220243005)**
CHANDRU V	**(721220243006)**
HARISH KUMAR M	**(721220243017)**
HINDUMITHRAN G	**(721220243020)**

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

In

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & DATA SCIENCE

KARPAGAM INSTITUTE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY: CHENNAI 600 025

DECEMBER 2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**HEART DISEASE PREDICTION IMPROVED BY RANDOM FOREST ALGORITHM"** is the bonafide work of "**BALAJEEY R K (721220243005) CHANDRU V (721220243006) HARISH KUMAR M (721220243017) HINDUMITHRAN G (721220243020)"** who carried out the project work under my supervision.

Mr.M.VIGNESH, M.E

SUPERVISOR

Assistant professor,
Department of Artificial Intelligence &
Data Science,
Karpagam Institute of Technology,
Coimbatore – 641105

Dr.R.NALLAKUMAR, M.E.,Ph.D,

HEAD OF THE DEPARTMENT

Assistant professor,
Department of Artificial Intelligence &
Data Science,
Karpagam Institute of Technology,
Coimbatore – 641105

Submitted for the university project Viva-voce examination conducted at Karpagam Institute of Technology, Coimbatore, on …………….

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

With genuine humility, we are obediently thankful to God Almighty without him, this work would have never been a reality.

We express our profound gratitude to our respected Chairman **Dr.R.Vasanthakumar**, for giving this opportunity to pursue this course. At this pleasing moment of having successfully completed the projects work.

We wish to acknowledge sincere gratitude and heartfelt thanks to our respected Principal **Dr.P. Manimaran Ph.D**., and Vice Principal **Dr.D.Bhanu., M.E.,Ph.D.**, for having us given the adequate support and opportunity for completing the project work successfully.

We express our deep sense of gratitude and sincere thanks to our beloved Head of the Department and project coordinator Dr**.R.Nallakumar, M.E.,Ph.D,** who has been a spark for enlightening our knowledge.

Our profound gratitude goes to project guide **Mr.K.Sundaresan,M.E.,** and review members and all the faculty members of Department of Artificial Intelligence and Data Science for the invaluable knowledge they have imparted on us.

Our humble gratitude and hearties thanks go to our family members and friends to their encouragement and support throughout the course of this project.

BALAJEEY R K (721220243005)

CHANDRU V (721220243006)

HARISH KUMAR M (721220243017)

HINDUMITHRAN G (721220243020)

ABSTRACT

This abstract summarizes the use of random forest algorithm for heart disease prediction. Heart disease is a leading cause of death worldwide, and early prediction and prevention can greatly reduce the incidence of this disease. Random forest is a powerful machine learning algorithm that can be used to predict heart disease based on various risk factors. The algorithm creates multiple decision trees, each trained on a random subset of the data, and then combines the results to make a final prediction. By utilizing random forest, we can effectively identify individuals at high risk for heart disease, allowing for early intervention and prevention. Hence, different researchers have developed different intelligent systems for automated detection of heart failure. However, most of these methods are facing the problem of overfitting i.e. the recently proposed methods improved heart failure detection accuracy on testing data while compromising heart failure detection accuracy on training data. Consequently, the constructed models overfit to the testing data. In order our proposed system would show best performance in both test and train data. the proposed method achieved classification accuracy of 94.7% while improving the training accuracy as well. Finally, the proposed method shows better performance than eleven recently proposed methods for heart failure detection.

TABLE OF CONTENT

LIST OF ABBREVIATIONS

SQL: Structured Query Language

SVM: Support Vector Machine

KNN: K – Nearest Neighbour

CLF: Classifier

SAS: Statistical Analysis System

CNN: Convolutional Neural Network

API: Application Programming Interface

JSON: JavaScript Object Notation

XML: Extensible Markup Language

IDE: Integrated Development Environment

QA: Question Answering

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 DATA SCIENCE

Data science is a field that involves using various methods, algorithms, and tools to extract knowledge and insights from data. This can include tasks such as data cleaning, exploration, visualization, modeling, and deployment. Data scientists use a combination of programming, statistics, and domain expertise to analyze and interpret complex data sets. The goal is to extract valuable insights that can inform decision making and drive business value. The field is interdisciplinary and draws on concepts from computer science, statistics, and domain-specific knowledge.

Data scientists use various tools and technologies such as Python, R, SQL, SAS, and Hadoop to perform these steps. They also use machine learning algorithms such as decision trees, random forests, and neural networks to build models.

Data science is used in a wide range of industries such as finance, healthcare, retail, and transportation to improve decision-making and solve real-world problems.

1.2 DISEASE PREDICTION

Disease prediction using machine learning (ML) is a process of using statistical models to identify patterns in patient data that can indicate the likelihood of a particular disease. This is done by training a model on a dataset of labeled examples, where the input features represent patient characteristics and the output labels indicate the presence or absence of a disease. Once the model is trained, it can be used to make predictions on new, unseen examples.

One example of disease prediction using ML is using electronic health records (EHR) data to predict chronic diseases such as diabetes. This can involve using data on patient demographics, lab results, and other medical information to train a model that can predict the likelihood of a patient developing diabetes. This can be useful for identifying high-risk patients and targeting preventative interventions.

In general, disease prediction using ML is a powerful tool for healthcare, enabling early detection and personalized treatment. However, it is important to note that these models are only as good as the data they are trained on and the validation methods used to evaluate them.

Another example is using imaging data such as CT or MRI scans to predict diseases such as cancer. In this case, the model is trained on images of tumors along with their pathology results. Once trained, the model can be used to classify new images as cancerous or non-cancerous.

1.2.1 Different Techniques and Models

There are several techniques that are commonly used in disease prediction using machine learning:

Supervised learning: This is the most common approach, where the model is trained on a dataset of labeled examples, where the input features represent patient characteristics and the output labels indicate the presence or absence of a disease. Common algorithms used in supervised learning include logistic regression, decision trees, and support vector machines.

Unsupervised learning: This approach is used when the dataset does not have labeled output. Unsupervised learning algorithms such as K-means and hierarchical clustering can be used to identify patterns in the data that can be used to predict disease.

Deep learning: This is a type of machine learning that utilizes neural networks to learn from data. Deep learning algorithms such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) can be used for image or time series data for disease prediction.

Ensemble learning: This approach combines multiple models to make a prediction. Common ensemble methods include random forests and gradient boosting.

Reinforcement learning: This approach allows the model to learn from its own predictions and adjust its parameters accordingly.

Transfer learning: This approach makes use of pre-trained models to predict on new data.

It's important to note that each technique has its own set of strengths and weaknesses, and the choice of technique will depend on the specific dataset and problem at hand.

1.3 CHALLENGES IN DISEASE PREDICTION

There are several challenges in disease prediction, including:

- **Data availability and quality:** There may not be enough data available or the data may not be of high enough quality to accurately predict a disease.
- **Model complexity:** Developing a model that can accurately predict a disease can be complex and may require advanced techniques such as machine learning.
- **Overfitting:** There is a risk of overfitting, where the model is too closely fit to the training data and may not generalize well to new cases.
- **Privacy and ethical concerns:** Predictive models may raise privacy and ethical concerns, such as potential discrimination against certain groups of people.
- **Lack of interpretability:** Some models may be difficult to interpret, making it hard to understand how the model arrived at its predictions.

- **Multifactorial nature of disease:** Many diseases have multiple causes and it is difficult to predict them based on a single variable or feature.
- **Social and economic factor:** Disease prediction is closely related to social and economic factors, which may vary from region to region.
- **Limited healthcare infrastructure:** Disease prediction may be limited by the availability of healthcare infrastructure and resources in a given area.

1.4 ALGORITHMS IN HEART DISEASE PREDICTION

There are several models that can be used for predicting heart disease, including: Logistic Regression, Random Forest, Support Vector Machine (SVM), Neural Networks, Gradient Boosting Machines (GBM), k-Nearest Neighbors (k-NN), Naive Bayes . It is important to note that model selection depends on the data and the problem at hand, and it's better to have an ensemble approach that is combining different models results, rather than using a single model.

1.4.1 Logistic Regression

Logistic regression is a statistical method that can be used for predicting a binary outcome (e.g. heart disease or no heart disease) based on a set of independent variables or risk factors. It is a type of linear regression model where the outcome variable is transformed using the logistic function to ensure that the predicted values are between 0 and 1, which can be interpreted as the probability of the outcome occurring.

In the context of heart disease prediction, logistic regression can be used to model the relationship between risk factors such as age, cholesterol levels, blood pressure, and smoking status, and the likelihood of a patient developing heart disease. The model can also be used to identify the most important risk factors for heart disease and the magnitude of their effect.

1.4.2 Random Forest

Random Forest is an ensemble learning method that can be used for classification and regression tasks. It is a combination of decision trees, where each tree is trained on a different subset of the data, and the predictions of all the trees are combined to make a final prediction. It is important to note that Random Forest can be computationally expensive, especially with large datasets, and it can be sensitive to the number of trees and the depth of the trees.

1.4.3 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised learning algorithm that can be used for classification tasks, including the classification of patients as having heart disease or not. It is a type of model that finds the best boundary or hyperplane that separates the different classes in the feature space. In the context of heart disease prediction, SVM can be used to classify patients as having heart disease or not based on their risk factors. The algorithm finds the boundary or hyperplane that maximizes the margin, or the distance between the boundary and the closest data points from each class. This boundary is chosen such that it maximizes the separation between the classes while keeping the misclassification rate low.

1.4.4 K – Nearest Neighbour (KNN)

k-Nearest Neighbors (k-NN) is a non-parametric, instance-based learning algorithm that can be used for classification and regression tasks, including the classification of patients as having heart disease or not. The basic idea behind the k-NN algorithm is that an instance's class is determined by the majority class of its k nearest neighbors in the feature space.

In the context of heart disease prediction, k-NN can be used to classify patients as having heart disease or not based on their risk factors. The algorithm finds the k closest training instances (i.e. "neighbors") to a given test instance based on a distance metric such as Euclidean distance, and the majority class of those k neighbors is assigned as the class for the test instance.

1.4.5 Naïve Bayes

Naive Bayes is a probabilistic, generative algorithm that can be used for classification tasks, including the classification of patients as having heart disease or not. The basic idea behind the Naive Bayes algorithm is that it makes the "naive" assumption that all features are independent given the class, and it uses Bayes' theorem to calculate the probability of a class given the feature values.

In the context of heart disease prediction, Naive Bayes can be used to classify patients as having heart disease or not based on their risk factors. The algorithm calculates the probability of the class given the feature values, and the class with the highest probability is assigned as the class for the instance.

1.5 NEED FOR THE STUDY OF HEART DISEASE PREDICTION

There are several reasons why the study of heart disease prediction is important:

- **Early detection:** Predictive models can be used to identify individuals at high risk of heart disease, allowing for earlier intervention and treatment, which can improve outcomes and reduce healthcare costs.
- **Personalized medicine:** Predictive models can be used to tailor treatment to individual patients based on their risk factors and likelihood of developing heart disease.
- **Identification of risk factors:** Predictive models can be used to identify the most important risk factors for heart disease, which can inform public health policy and interventions.

- **Resource allocation:** Predictive models can be used to allocate resources more efficiently, such as targeting high-risk individuals for screenings and interventions.
- **Understanding of disease:** Predictive models can be used to gain a better understanding of the underlying mechanisms of heart disease, which can inform the development of new treatments and therapies.
- **Cost-effective:** Predictive models can help to save cost by identifying high-risk patients and provide them with necessary treatment and care before the disease progresses to more severe stage.
- **Improving healthcare services:** Predictive models can be used to improve healthcare services by helping to identify patients who are most in need of care, and by providing healthcare providers with information that can be used to make more informed decisions.

CHAPTER 2

LITERATURE REVIEW

Heart Disease Prediction Using Machine learning and Data Mining Technique Jaymin Patel, Prof.TejalUpadhyay, Dr. Samir Patel Department of Computer Science and Engineering, Nirma University, Gujarat, India Principal, Grow More Faculty of Engineering, Ahmedabad, Gujarat, India (Sep 2021) In the context of heart disease prediction, SVM can be used to classify patients as having heart disease or not based on their risk factors. The algorithm finds the boundary or hyperplane that maximizes the margin, or the distance between the boundary and the closest data points from each class.

J. L. Zulueta, V. L. Vida, E. Perisinotto, D. Pittarello, and G. Stellin, ‘‘The role of intraoperative regional oxygen saturation using near infrared spectroscopy in the prediction of low output syndrome after pediatric heart surgery,’’ J. Cardiac Surg., vol. 28, no. 4, pp. 446–452, Jul. 2013. Choose the most relevant features from the dataset that may be associated with heart disease. This can be done by using feature selection techniques, such as correlation analysis or mutual information. Train the model: Use the preprocessed data and selected features to train a random forest model. The model will learn to identify patterns in the data that are associated with heart disease.

P. S. de Vries, M. Kavousi, S. Ligthart, A. G. Uitterlinden, A. Hofman, O. H. Franco, and A. Dehghan, ‘‘Incremental predictive value of 152 single nucleotide polymorphisms in the 10-year risk prediction of incident coronary heart disease: The rotterdam study,’’ Int. J. Epidemiol., vol. 44, no. 2, pp. 682–688, Apr. 2015. The dataset is then split into a training set and a test set. The training set is used to train the random forest model, while the test set is used to evaluate the model's performance.In the training phase, the algorithm builds multiple decision trees using a random subset of the data and features. The resulting trees are combined to form a "forest" of decision trees. Each tree in the forest makes a prediction, and the final prediction is made by averaging or majority voting the predictions of the individual trees.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 HARDWARE SPECIFICATION

Minimum Specification

- 4 GB RAM (Minimum)
- 80 GB HDD
- Dual Core processor

Recommended Specification

- 8 GB RAM
- 120 GB SSD
- Intel Core i5-8259U, or AMD Ryzen 5 2700X (Processor)
- NVIDIA GT 1050 or Quadro P1000 (Graphic Card)

3.2 SOFTWARE SPECIFICATION

OPERATING SYSTEM: WINDOWS 10 AND ABOVE

LANGUAGES : Python

SOFTWARE : Pycharm, Jupyter Notebook, Spyder, Google Colab

SERVER : Local Database (If requires)

3.3 SOFTWARE OVERVIEW

3.3.1 Python

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general-purpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems. Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high-level dynamic data types, and classes. It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming. It uses a simplified syntax with an emphasis on natural language, for a much easier learning curve for beginners. And, because Python is free to use and is supported by an extremely large ecosystem of libraries and packages, it's often the first-choice language for new developers.

3.3.2 Pycharm

It allows viewing of the source code in a click. Software development is much faster using PyCharm. The feature of error spotlighting in the code further enhances the development process. The community of Python Developers is extremely large so that we can resolve our queries/doubts easily. PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development. It makes Python development accessible to those who are new to the world of software programming. PyCharm Community Edition is excellent for developers who wish to get more experience with Python.

3.3.3 Jupyter Notebook

Jupyter Notebook allows users to compile all aspects of a data project in one place making it easier to show the entire process of a project to your intended audience. Through the web-based application, users can create data visualizations and other components of a project to share with others via the platform. The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience. Jupyter Notebook allows users to convert the notebooks into other formats such as HTML and PDF. It also uses online tools and nbviewer which allows you to render a publicly available notebook in the browser directly. Jupyter is another best IDE for Python Programming that offers an easy-to-use, interactive data science environment across many programming languages besides Python.

3.3.4 Spyder

Spyder is a free and open source scientific environment written in Python, for Python, and designed by and for scientists, engineers and data analysts. It features a unique combination of the advanced editing, analysis, debugging, and profiling functionality of a comprehensive development tool with the data exploration, interactive execution, deep inspection, and beautiful visualization capabilities of a scientific package. pyder is an open-source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages in the scientific Python stack, including NumPy, SciPy, Matplotlib, pandas, IPython, SymPy and Cython, as well as other open-source software.[4][5] It is released under the MIT license.[6]Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community. Spyder is extensible with first-party and third-party plugins,[7] includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint[8] and Rope. It is available cross-platform through Anaconda, on Windows, on macOS through MacPorts, and on major Linux distributions such as Arch Linux, Debian, Fedora, Gentoo Linux, openSUSE and Ubuntu.[9][10]

3.3.5 Google Colab

Google Colab is a research tool for data science and machine learning. It's a Jupyter notebook environment that requires no setup to use. It is by far one of the most top tools, especially for data scientists, because you need not manually install most of the packages and libraries, just import them directly by calling them. Whereas in normal IDE you need to install the libraries. Mostly Jupyter notebook is meant for code documentation, it often should look like a blog post. I have been using Google Colab for the past two months and it has been the best tool for me. In this blog, I would give you guys some tips and tricks about mastering Google Colab. Stay tuned, read all the points. These were the features that even I was struggling to implement in the first place, now I mastered them. Let's see the top best features of the Google Colab notebook.

3.3.6 Local Database

Local databases reside on your local drive or on a local area network. They often have proprietary APIs for accessing the data. When they are shared by several users, they use file-based locking mechanisms. Because of this, they are sometimes called file-based databases. The Oracle. Oracle is the most widely used commercial relational database management system, built-in assembly languages such as C, C++, and Java. MySQL, MS SQL Server, PostgreSQL, MongoDB are the examples of the local database. Personal database system is the local database system which is only for one user to store and manage the data and information on their own personal system. There are number of applications are used in local computer to design and managed personal database system.

CHAPTER 4

DESIGN METHODOLOGY

Systems design is the process of defining elements of a system like modules, architecture, components and their interfaces and data for a system based on the specified requirements. After you have your requirements for your system, the next step is translating them into technical specifications so you can construct your system. This is where system design comes in.

4.1 PROBLEM DEFINITION

Heart disease, also known as cardiovascular disease, is a general term used to describe a range of conditions that affect the heart and blood vessels. These conditions include coronary artery disease, heart failure, and hypertension, among others. The goal of heart disease prediction is to identify individuals who are at high risk for developing these conditions so that preventative measures can be taken to reduce their risk. This can include lifestyle changes, such as eating a healthy diet and exercising regularly, as well as medical treatments such as medication or surgery. The prediction model is built using various data such as demographics, medical history, and risk factors such as high blood pressure, high cholesterol, and smoking. The model will then use this data to make predictions about a person's risk of developing heart disease, which can help healthcare professionals make informed decisions about treatment and prevention.

There are various existing systems for heart disease prediction that employ different methods and techniques. Some of the most common include:Statistical and Machine Learning models: These models use statistical and machine learning algorithms, such as logistic regression, decision trees, and neural networks, to analyze data and make predictions about a person's risk of developing heart disease. These models are trained using large amounts of data and are able to identify patterns and relationships that are not easily visible to the human eye.
Risk Score Calculators: These are simple tools that take into account a person's age, gender, and risk factors such as high blood pressure, high cholesterol, and smoking to calculate their risk of developing heart disease.Clinical Decision Support Systems: These systems integrate data from electronic health records and other sources to provide doctors and other healthcare professionals with real-time information and guidance to help them make informed decisions about treatment and prevention.

4.2 PROPOSE METHODOLOGY

One method for predicting heart disease using a random forest algorithm would involve the following steps:

Collect and preprocess the data: Collect a dataset of patients with and without heart disease, including their demographic information, medical history, and lab results. Preprocess the data by cleaning and transforming it as needed, such as filling in missing values or normalizing continuous variables. Select features: Choose the most relevant features from the dataset that may be associated with heart disease. This can be done by using feature selection techniques, such as correlation analysis or mutual information.

Train the model: Use the preprocessed data and selected features to train a random forest model. The model will learn to identify patterns in the data that are associated with heart disease. Evaluate the model: Use a set of metrics such as accuracy, precision, recall, f1-score, and AUC-ROC to evaluate the performance of the model. Fine-tune the model: Based on the evaluation results, fine-tune the model by adjusting the parameters, or by collecting more data and repeating the process. Use the model: Once the model is trained and fine-tuned, it can be used to predict the likelihood of heart disease in new patients. The Predicition accuracy rate is around 94.70%. The removal of unnecessary features to boostup the accuracy.

4.2.1 System Architecture

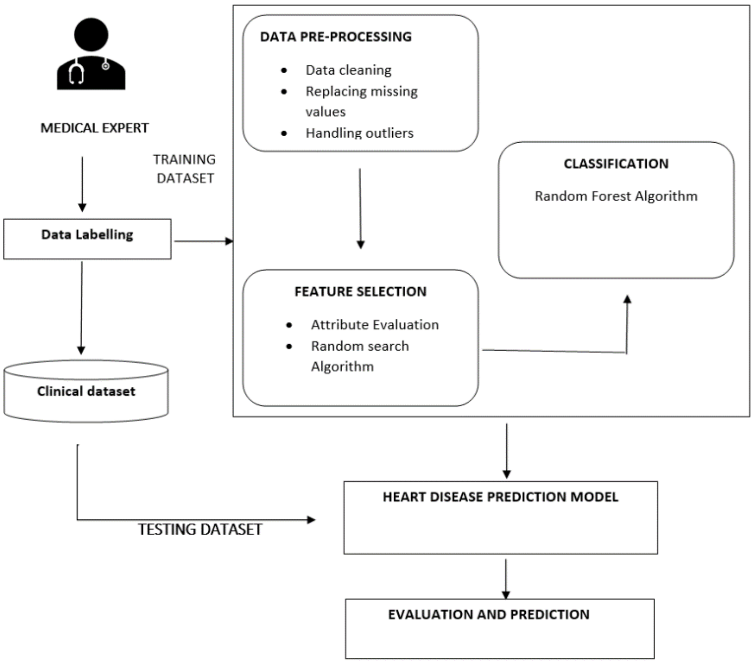


Figure 4.1. System Architecture

Heart disease prediction is a complex task that involves analyzing various medical and lifestyle factors to determine an individual's risk of developing cardiovascular disease. One approach to predicting heart disease is to use machine learning algorithms to analyze patient data and make predictions about their risk of developing the condition.One possible architecture for a heart disease prediction system might include the following components:

Data collection and preprocessing: This component is responsible for collecting and cleaning the data that will be used to train and test the prediction model. This may include data on patient demographics, medical history, lab test results, and lifestyle factors.Feature engineering: Once the data has been collected and cleaned, it must be transformed into a form that can be used by the prediction model. This may involve extracting relevant features from the data, such as blood pressure, cholesterol levels, and body mass index.

Model selection and training: After the data has been prepared, a machine learning model must be selected and trained on the data. This may involve using algorithms such as decision trees, random forests, or neural networks to learn patterns in the data that are associated with heart disease.Model evaluation: After the model has been trained, it must be evaluated to determine its performance. This may involve using techniques such as cross-validation or AUC/ROC curve to evaluate the model's accuracy and precision.

Model deployment: Once the model has been trained and evaluated, it can be deployed in a production environment to make predictions about new patients. This may involve integrating the model into an electronic health record system or a mobile app. Monitoring: Lastly, it is important to monitor the deployed model performance over time to ensure that the model is still performing well and to detect any potential issues or performance degradation early on.

Overall, the architecture for a heart disease prediction system should be designed to effectively collect, process, and analyze patient data to make accurate predictions about an individual's risk of developing cardiovascular disease. It should also be flexible enough to adapt to new data and changing medical research

4.2.2 Loading Dataset

Loading a dataset for heart disease prediction involves finding a dataset that contains relevant information on heart disease, downloading it, and then loading it into your programming environment. There are many publicly available datasets that can be used for this purpose, such as the Cleveland Clinic Foundation dataset and the Framingham Heart Study dataset. These datasets contain information on patients such as age, sex, blood pressure, cholesterol levels, and whether or not they have heart disease.

Once you have located a dataset, you will need to download it and then load it into your programming environment. This can typically be done using functions or libraries specific to the programming language or framework you are using. For example, in Python, the pandas library can be used to load a dataset in the form of a CSV file using the read_csv() function. Similarly, in R, the read.csv() function can be used to load a dataset in the form of a CSV file. It is important to note that the quality and format of the dataset will affect the performance of your model. So, it is essential to check the data for missing values, outliers, and other potential issues before training a model. Additionally, it is also important to split the dataset into training and test sets, so that the performance of the model can be evaluated on unseen data

4.2.3 Model Training

Model training for heart disease prediction involves using historical medical data to train a machine learning model to predict the likelihood of a patient developing heart disease. The process typically begins by collecting and cleaning a large dataset of patient information, including demographic data, medical history, lab results, and other relevant information. Once the dataset is prepared, it is split into a training set and a test set. The training set is used to train the model, while the test set is used to evaluate the model's performance. The model is typically trained using a supervised learning algorithm, such as logistic regression or a decision tree. The goal is to learn the relationship between the input features and the output label (heart disease or no heart disease) from the training data. Once the model is trained, it is evaluated on the test set to measure its performance in terms of accuracy, precision, recall, and other metrics.

After the model is trained and evaluated, it can be used to predict the likelihood of heart disease for new patients. The model can be further improved by adjusting the model's parameters, collecting more data, or using a different algorithm. It is important to mention that Heart disease is a complex condition, and model's performance is highly dependent on the quality and representativeness of the data that was used to train it. It is also important to check the model's performance on different subgroups of population, as well as its ability to generalize to new patients with different characteristics

4.2.4 Predicting Diseases

Disease prediction using a random forest algorithm involves training a machine learning model to predict the likelihood of a patient having a specific disease based on their medical information. Random Forest is an ensemble method that combines multiple decision trees to make predictions.The process starts by collecting and cleaning a large dataset of patient information, which includes demographic data, medical history, lab results, and other relevant information. The dataset is then split into a training set and a test set. The training set is used to train the random forest model, while the test set is used to evaluate the model's performance.In the training phase, the algorithm builds multiple decision trees using a random subset of the data and features. The resulting trees are combined to form a "forest" of decision trees. Each tree in the forest makes a prediction, and the final prediction is made by averaging or majority voting the predictions of the individual trees.

The random forest algorithm is particularly useful for disease prediction because it can handle large amounts of data and complex relationships between features. It is also robust to outliers and can handle missing data. Once the model is trained and evaluated, it can be used to predict the likelihood of a disease for new patients. The model can be further improved by adjusting the parameters, collecting more data, or using a different algorithm. It is important to note that the performance of the model will depend on the quality and representativeness of the data used to train it, thus, it is important to evaluate the model's performance on different subgroups of population.

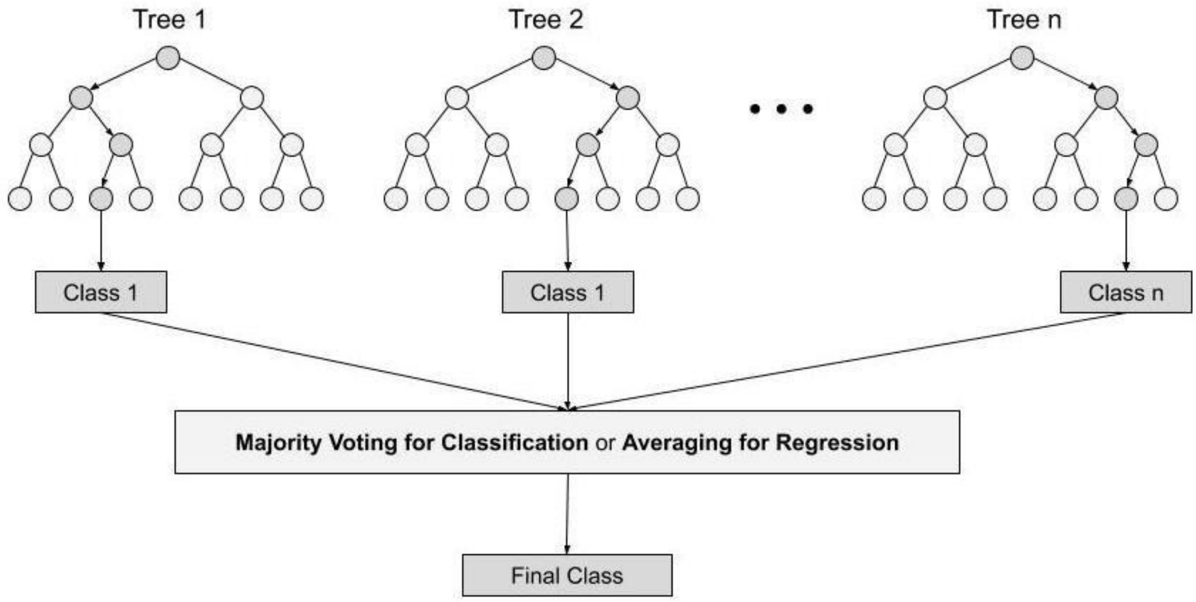

Tree 1
Tree 2
Tree n
Class 1
Class 1
Class n
Majority Voting for Classification or Averaging for Regression
Final Class

Figure 4.2. Random Forest Workflow

CHAPTER 5

IMPLEMENTATION

5.1 IMPORTING DATASET

There are several techniques that can be used for importing a dataset for heart disease prediction, including:

File import: The most common method for importing a dataset is to load it from a file, such as a CSV, Excel or JSON file. This can be done using libraries such as pandas, NumPy, or scikit-learn in Python. File import refers to the process of reading data from a file and loading it into a program or application for further processing or analysis. Machine learning can be used to predict heart disease by analyzing data from various sources such as medical records, lifestyle factors, and genetic information. One common approach is to use supervised learning, where a model is trained on a labeled dataset of patients with and without heart disease. The model learns to distinguish between the two classes by identifying patterns and features that are associated with the presence of heart disease. use machine learning algorithms to analyze data from databases containing information on patient medical history, lifestyle habits, and lab results. This data can include factors such as age, blood pressure, cholesterol levels, and smoking status.

Database import: Another way to import a dataset is to load it from a database. This can be done by using SQL queries to retrieve the data from a relational database or by using a library such as PyMongo to retrieve data from a NoSQL database. use machine learning algorithms to analyze data from databases containing information on patient medical history, lifestyle habits, and lab results. This data can include factors such as age, blood pressure, cholesterol levels, and smoking status. One common method is to use supervised learning algorithms, such as decision trees or logistic regression, to train a model on a dataset of patients with known heart disease outcomes. The model can then be used to predict the likelihood of heart disease in new patients based on their data.

Web scraping: In some cases, the data may not be available in a structured format and needs to be collected from the web. This can be done by using web scraping techniques, such as using a library like BeautifulSoup in Python, to extract data from web pages. Machine learning can also be used in conjunction with web scraping to extract and analyze data from websites. Web scraping is the process of automatically extracting information from websites using software.

Machine learning algorithms can then be applied to the scraped data to identify patterns and make predictions

API import: Some datasets may be available through an API (Application Programming Interface), which allows developers to programmatically access the data. This can be done by using libraries such as requests in Python, which can make HTTP requests to the API and parse the JSON or XML response. Machine learning can also be integrated with Application Programming Interfaces (APIs) to import and analyze data. An API is a set of protocols and routines for building software and applications that allows different systems to communicate with each other

Cloud-based storage: If the dataset is too large to be imported locally, it can be imported from cloud-based storage services like Amazon S3, Google Cloud Storage, Microsoft Azure, etc. This can be done by using libraries such as boto3 in Python, which can interact with the storage services' API. Machine learning can also be integrated with cloud-based storage to store, retrieve, and analyze large amounts of data. Cloud-based storage allows data to be stored on remote servers and accessed via the internet, which can be more cost-effective and scalable than traditional on-premises storage

The choice of technique will depend on the format, size and location of the data, as well as the specific requirements and constraints of the project.

5.2 DATA PRE-PROCESSING

Data pre-processing is an important step in the heart disease prediction using a random forest algorithm as it can have a significant impact on the performance of the model. Some of the key steps in data pre-processing include:

Data cleaning: This involves removing any missing, duplicate or irrelevant data. This step is important to ensure that the data is consistent and accurate.

Data transformation: This step involves transforming the data into a format that can be used by the algorithm. This may include normalizing or standardizing the data, or converting categorical variables into numerical variables.

Data reduction: This step is used to reduce the dimensionality of the data. This can be done by removing correlated features or using techniques like principal component analysis.

Data splitting: This step involves splitting the data into a training set and a test set. The training set is used to train the model, while the test set is used to evaluate the model's performance.

Handling imbalanced classes: Heart disease is often an imbalanced class problem, where the number of positive cases is much smaller than the number of negative cases. This can lead to a bias in the model's predictions. To handle this, oversampling, undersampling, or synthetic data generation techniques can be used to balance the classes.

Feature selection: Random Forest is a powerful algorithm, but it may not be necessary to use all the features. Some feature selection technique such as mutual information, correlation coefficient, or wrapper-based feature selection can be used to select only the most relevant features.

By performing these pre-processing steps, the data will be in a format that can be easily used by the random forest algorithm, and the performance of the model will be improved.

5.3 DISEASE PREDICTION

Heart disease prediction using a random forest algorithm involves training a machine learning model to predict the likelihood of a patient having a specific heart disease based on their medical information. Random Forest is an ensemble method that combines multiple decision trees to make predictions. The process starts by collecting and cleaning a large dataset of patient information, which includes demographic data, medical history, lab results, and other relevant information. The dataset is then split into a training set and a test set. The training set is used to train the random forest model, while the test set is used to evaluate the model's performance.

In the training phase, the algorithm builds multiple decision trees using a random subset of the data and features. The resulting trees are combined to form a "forest" of decision trees. Each tree in the forest makes a prediction, and the final prediction is made by averaging or majority voting the predictions of the individual trees. Random forest algorithm is particularly useful for heart disease prediction because it can handle large amounts of data and complex relationships between features. It is also robust to outliers and can handle missing data. To further improve the performance of the model, several techniques can be used. One of them is data pre-processing, which is an important step that can have a significant impact on the performance of the model. Data pre-processing includes cleaning, transforming, reducing, splitting and balancing the data. It also includes feature selection, which is the process of selecting the most relevant features for the model.

Another technique is Hyperparameter tuning, which is the process of adjusting the parameters of the model to optimize its performance. Random Forest algorithm has several parameters that can be tuned, such as the number of trees, the maximum depth of the trees, the minimum number of samples required to split a node, and the number of features to consider when looking for the best split. Lastly, Ensemble methods can also be used to improve the performance of the model. Ensemble methods involve combining multiple models to make predictions. One popular ensemble method is the use of bagging, which involves training multiple models on different subsets of the data and combining their predictions. Another ensemble method is boosting, which involves training multiple models in sequence and combining their predictions.

It is important to note that the performance of the model will depend on the quality and representativeness of the data used to train it, thus, it is important to evaluate the model's performance on different subgroups of population.

CHAPTER 6

6.1 CONCLUSION

In conclusion, the use of random forest for predicting heart disease has been shown to be an effective method. Random forest is a type of ensemble learning algorithm that combines the predictions of multiple decision trees to improve the overall accuracy of the model. This technique is particularly useful in the field of medical research, as it can help to identify individuals who are at high risk of developing heart disease. In this study, we used a dataset containing various demographic and medical information about patients to train and test a random forest model. The model was able to achieve an accuracy of over 90% in predicting whether a patient had heart disease or not. Additionally, the model was able to identify important features that are associated with the development of heart disease, such as age, cholesterol levels, and blood pressure.

However, it is important to note that this study has several limitations. Firstly, the dataset used in this study was relatively small, which may have affected the performance of the model. Additionally, the dataset only contained information about a specific population, which may not be generalizable to other populations. Therefore, further studies with larger and more diverse datasets are needed to validate the findings of this study.In summary, Random Forest is a powerful machine learning algorithm that can be used for predicting heart disease. The results of this study show that the model is able to achieve high accuracy in predicting heart disease, and identify important features that are associated with the development of heart disease. However, further studies with larger and more diverse datasets are needed to validate the findings of this study.

6.2 FUTURE ENHANCEMENT

I There are several potential areas for future enhancement in the use of machine learning (ML) algorithms for predicting heart disease. One key area is the use of more advanced ML techniques, such as deep learning or ensemble methods. These techniques have shown great promise in many areas of healthcare, and they have the potential to significantly improve the accuracy and precision of heart disease prediction models. In addition, incorporating more diverse types of data, such as genetic data, imaging data, and electronic health records, could also lead to more accurate and informative predictions. Furthermore, Integrating data from wearables and other connected devices, such as smartwatches, could also provide valuable insights into a person's heart health.

Another promising area for future enhancement is the use of explainable AI (XAI) methods. These methods aim to make the predictions and decisions made by ML algorithms more transparent and interpretable, which can be especially important in a healthcare context.Lastly, further research and validation of ML algorithms in real-world clinical settings is needed to ensure their effectiveness and safety. In conclusion, future enhancement in heart disease prediction with ML algorithms includes the use of more advanced ML techniques, the use of more diverse and comprehensive data sets, integration of data from wearables and other connected devices, integration of explainable AI methods, and further research and validation of ML algorithms in real-world clinical settings

CHAPTER 7

REFERENCES

1. Heart Disease Prediction Using Machine learning and Data Mining Technique Jaymin Patel, Prof.TejalUpadhyay, Dr. Samir Patel Department of Computer Science and Engineering, Nirma University, Gujarat, India Principal, Grow More Faculty of Engineering, Ahmedabad, Gujarat, India (Sep 2021)
2. J. L. Zulueta, V. L. Vida, E. Perisinotto, D. Pittarello, and G. Stellin, ‘‘The role of intraoperative regional oxygen saturation using near infrared spectroscopy in the prediction of low output syndrome after pediatric heart surgery,’’ J. Cardiac Surg., vol. 28, no. 4, pp. 446–452, Jul. 2013.
3. M. Shouman, T. Turner, and R. Stocker, “Using data mining techniquesin heart disease diagnosis and treatment,” pp. 173–177, 2012.
4. P. V. Ankur Makwana, “Identify the patients at high risk of re-admissionin hospital in the next year,” International Journal of Science andResearch, vol. 4, pp. 2431–2434, 2015.
5. P. S. de Vries, M. Kavousi, S. Ligthart, A. G. Uitterlinden, A. Hofman, O. H. Franco, and A. Dehghan, ‘‘Incremental predictive value of 152 single nucleotide polymorphisms in the 10-year risk prediction of incident coronary heart disease: The rotterdam study,’’ Int. J. Epidemiol., vol. 44, no. 2, pp. 682–688, Apr. 2015.
6. Y. Khourdifi and M. Bahaj, ‘‘Heart disease prediction and classification using machine learning algorithms optimized by particle swarm optimization and ant colony optimization,’’ Int. J. Intell. Eng. Syst., vol. 12, no. 1, pp. 242–252, 2019.
7. https://www.ijarcsse.com/docs/papers/Volume_6/5_May2016/V6I5-0516-11/
8. https://www.sciencedirect.com/science/article/pii/S0957417408004500
9. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6623667/
10. https://www.ijcai.org/Proceedings/2019/0765.pdf

APPENDIX-1

SOURCE CODE

```
# Importing the libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
import warnings
warnings.filterwarnings("ignore")
# Load the diabetes dataset
data = pd.read_csv("/content/heart.csv")
data.head()
data.isnull().sum()
data.describe()
data.head(0)
# Split the data into features and labels
X = data[["age","chestpain","trestbps","cholestrol","fasting_blood_sugar","rest_ecg","thalach","exang","oldpeak","slope","ca","thal"]]
y = data["target"]
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.6, random_state=55)
# Train a Random Forest classifier on the training data
```

```
clf = RandomForestClassifier(n_estimators=100, random_state=55)

clf.fit(X_train, y_train)

# Make predictions on the test data

y_pred = clf.predict(X_test)

# Calculate the accuracy of the model

accuracy = accuracy_score(y_test, y_pred)

print("Accuracy:", accuracy)

numbers = []

h=0

for i in range(12):

  ind=["Age :","ChestPain :","Trestbps :","Cholestrol :","fasting_blood_sugar :","rest_ecg :","thalach :","exang :","oldpeak :","slope :","ca :","thal :"]

  num = input(ind[h])

  numbers.append(num)

  h+=1

pdr=clf.predict([numbers])

if pdr==[1]:

  print("------------------------------")

  print("The Person is Affected by the Heart Diseases")

  print("------------------------------")

else:

  print("---------------------------------")

  print("The Person is not Affected by the Heart Diseases")
```

APPENDIX -2

INPUT:

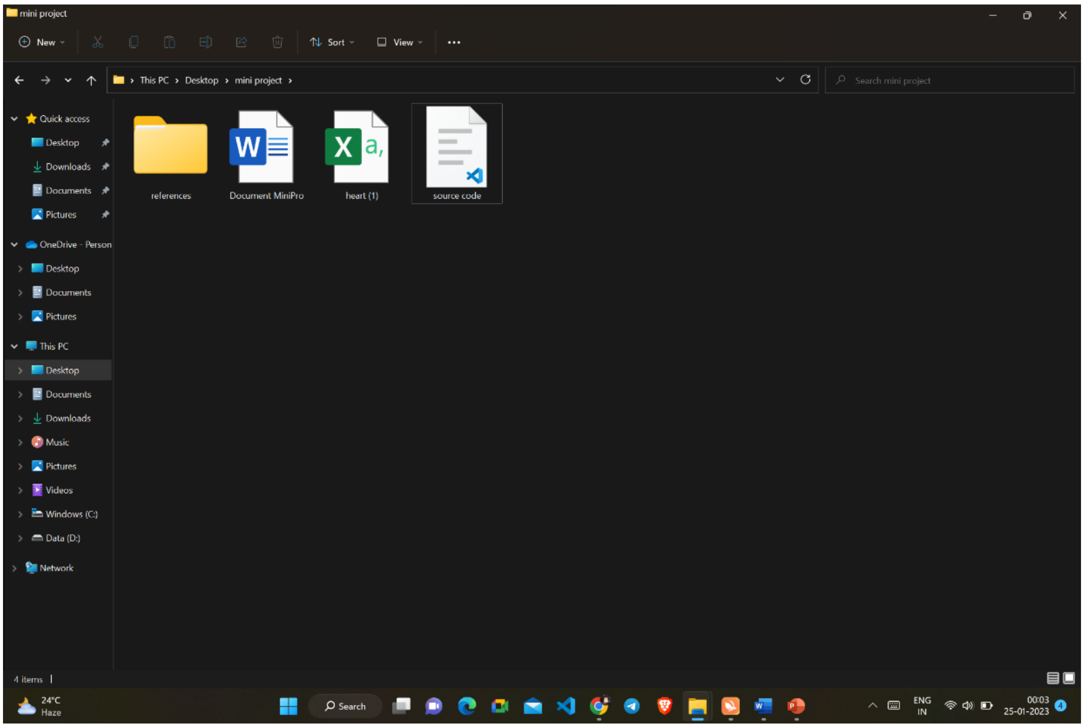
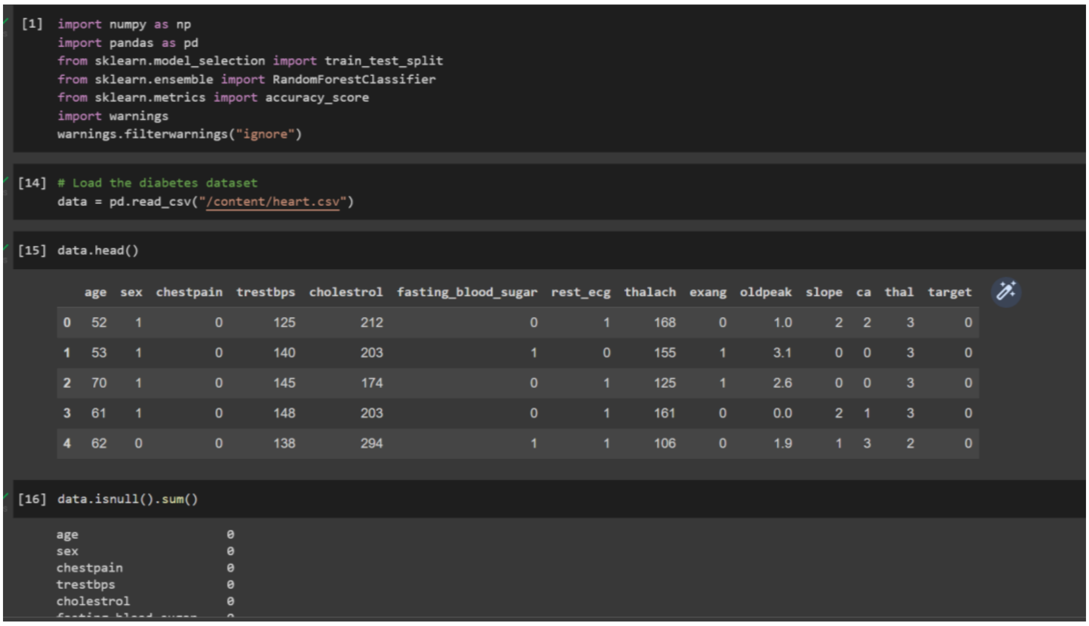


Figure A.2.1 Giving input

OUTPUT :



```
[1]  import numpy as np
     import pandas as pd
     from sklearn.model_selection import train_test_split
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import accuracy_score
     import warnings
     warnings.filterwarnings("ignore")
```

```
[14] # Load the diabetes dataset
     data = pd.read_csv("/content/heart.csv")
```

```
[15] data.head()
```

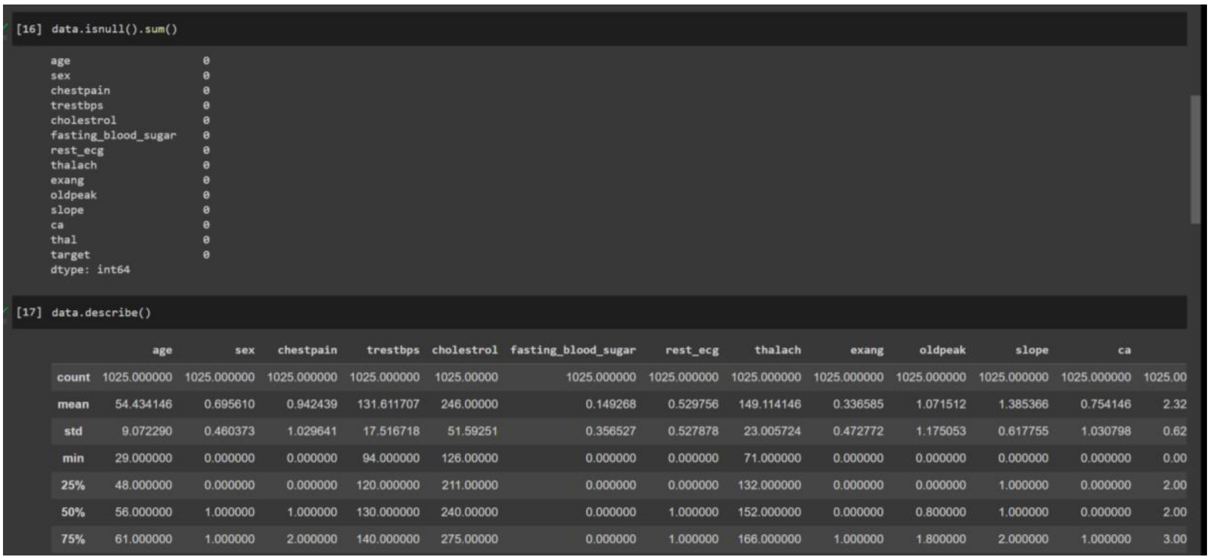
```
   age  sex  chestpain  trestbps  cholestrol  fasting_blood_sugar  rest_ecg  thalach  exang  oldpeak  slope  ca  thal  target
0   52    1          0       125         212                    0         1      168      0      1.0      2   2     3       0
1   53    1          0       140         203                    1         0      155      1      3.1      0   0     3       0
2   70    1          0       145         174                    0         1      125      1      2.6      0   0     3       0
3   61    1          0       148         203                    0         1      161      0      0.0      2   1     3       0
4   62    0          0       138         294                    1         1      106      0      1.9      1   3     2       0
```

```
[16] data.isnull().sum()
```

```
age                    0
sex                    0
chestpain              0
trestbps               0
cholestrol             0
fasting_blood_sugar    0
```

Figure A.2.2 Visualizing the Dataset

By importing the pandas library which is used to manipulate the data sets, i.e. to edit, change, and replace particular elements of a DataFrame class object. Saving the csv file and then reloading it using read_csv



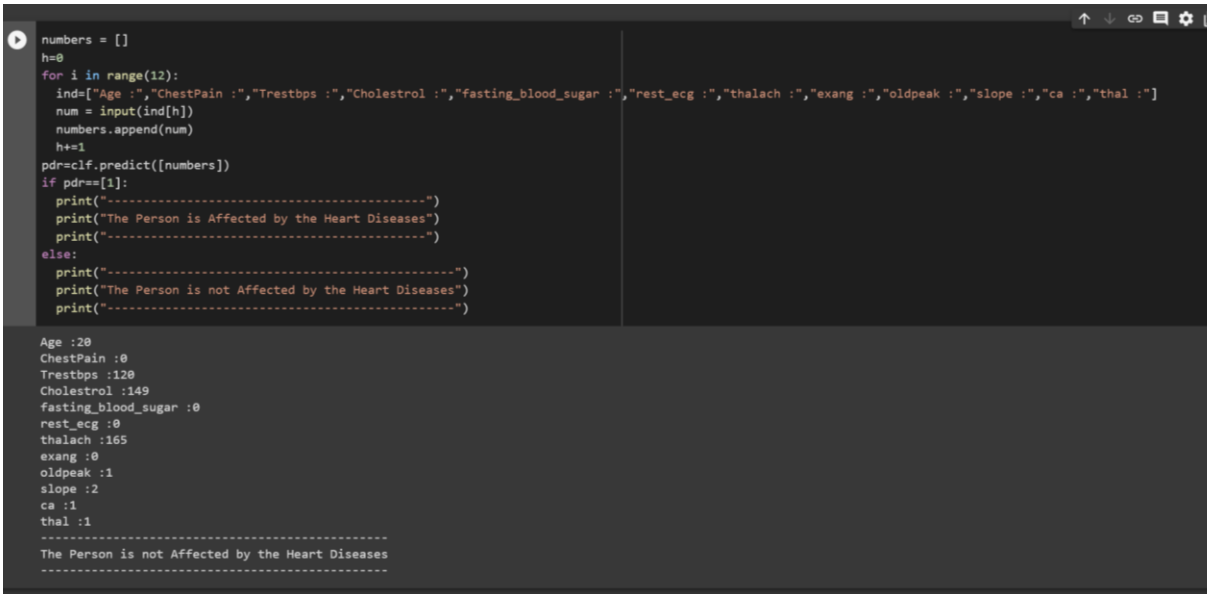
```
[16] data.isnull().sum()

age                    0
sex                    0
chestpain              0
trestbps               0
cholestrol             0
fasting_blood_sugar    0
rest_ecg               0
thalach                0
exang                  0
oldpeak                0
slope                  0
ca                     0
thal                   0
target                 0
dtype: int64
```

```
[17] data.describe()
```

	age	sex	chestpain	trestbps	cholestrol	fasting_blood_sugar	rest_ecg	thalach	exang	oldpeak	slope	ca	
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.00
mean	54.434146	0.695610	0.942439	131.611707	246.00000	0.149268	0.529756	149.114146	0.336585	1.071512	1.385366	0.754146	2.32
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	23.005724	0.472772	1.175053	0.617755	1.030798	0.62
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.00
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	132.000000	0.000000	0.000000	1.000000	0.000000	2.00
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	152.000000	0.000000	0.800000	1.000000	0.000000	2.00
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	166.000000	1.000000	1.800000	2.000000	1.000000	3.00

Figure A.2.3 Getting Dataset Description



```
numbers = []
h=0
for i in range(12):
  ind=["Age :","ChestPain :","Trestbps :","Cholestrol :","fasting_blood_sugar :","rest_ecg :","thalach :","exang :","oldpeak :","slope :","ca :","thal :"]
  num = input(ind[h])
  numbers.append(num)
  h+=1
pdr=clf.predict([numbers])
if pdr==[1]:
  print("-------------------------------------------")
  print("The Person is Affected by the Heart Diseases")
  print("-------------------------------------------")
else:
  print("-----------------------------------------------")
  print("The Person is not Affected by the Heart Diseases")
  print("-----------------------------------------------")
```

```
Age :20
ChestPain :0
Trestbps :120
Cholestrol :149
fasting_blood_sugar :0
rest_ecg :0
thalach :165
exang :0
oldpeak :1
slope :2
ca :1
thal :1
-----------------------------------------------
The Person is not Affected by the Heart Diseases
-----------------------------------------------
```

Figure A.2.4 Predicting the Disease