**HEART DISEASE PREDICTION**

**CS3361 DATA SCIENCE LABORATORY**

**A MINIPROJECT REPORT**

*Submitted by*

**DEEPTHI PON GAYATHRI P (953622104017)**

**RATHINA BHARGHAVI A (953622104082)**

**VIJAYALAKSHMI T (953622104118)**

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**RAMCO INSTITUTE OF TECHNOLOGY,**

**RAJAPALAYAM**

**JANUARY 2024**

**BONAFIDE CERTIFICATE**

Certified that this mini project report **“HEART DISEASE PREDICTION”** is the bonafide work of **“DEEPTHI PON GAYATHRI P (953622104017), RATHINA BHARGHAVI A (953622104082), VIJAYALAKSHMI T (953622104118)”** who carried out the miniproject work under my supervision.

|  |  |
| --- | --- |
| **SIGNATURE**  Mr.K.VigneshSaravanan B.E.,M.Tech.  **FACULTY IN-CHARGE**  Assistant Professor  Department of Computer Science  and Engineering.  Ramco Institute of Technology,  North Venganallur Village,  Rajapalayam – 626117 | **SIGNATURE**  Dr.K.Vijayalakshmi M.E., Ph.D.  **HEAD OF THE DEPARTMENT**  Department of Computer Science and Engineering.  Ramco Institute of Technology,  North Venganallur Village,  Rajapalayam– 626117 |

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ABSTRACT**

In today's world, maintaining good health is crucial for everyone, and predicting the risk of heart disease has become a necessity to enhance our well-being. Understanding our health status helps us make informed decisions and adopt a healthier lifestyle. To facilitate this, heart disease prediction systems prove to be invaluable, drawing insights from various data sources to generate personalized risk assessments for individuals. These prediction systems find applications in diverse sectors such as healthcare, insurance, and wellness. They leverage different methodologies to analyze health-related data and provide accurate predictions. One common approach involves examining the medical history and health metrics of individuals, comparing them to similar cases, and identifying potential risk factors. Similar to movie recommendation systems, hybrid approaches are often employed in heart disease prediction systems. These hybrids combine different techniques, incorporating both historical health data and machine learning algorithms to enhance prediction accuracy. By considering a person's past medical history and comparing it with a broader dataset of similar cases, these systems can offer more robust predictions. The popularity of a particular health prediction model depends on its ability to provide accurate assessments and recommendations. Reviews and feedback from healthcare professionals and users contribute significantly to the system's credibility. Positive experiences and successful predictions are likely to influence others to adopt the system for preventive healthcare measures. Analyzing the sentiment behind reviews and testimonials becomes crucial in this context as well. Sentiment analysis helps filter out misleading information and ensures that the predictions are based on reliable data. Users are more likely to trust a heart disease prediction system that has garnered positive reviews and has a track record of providing valuable insights.

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CONTENT** | **PAGE** **NO** |  |
| **1.** | **INTRODUCTION** | 1 |  |
|  | 1.1 Heart Disease Prediction | 1 |  |
|  | 1.2 Project objectives | 2 |  |
|  | 1.3 Project specification | 2 |  |
| **2.** | **SYSTEM SPECIFICATION** | 3 |  |
|  | 2.1 Hardware specification | 3 |  |
|  | 2.2 Software specification | 3 |  |
| **3.** | **PACKAGES** | 4 |  |
|  | 3.1 Numpy | 4 |  |
|  | 3.2 Pandas | 5 |  |
|  | 3.3 Matplotlib | 7 |  |
|  | 3.3.1 Matplotlib bar plot | 8 |  |
|  | 3.3.2 Matplotlib histogram | 9 |  |
| **4.** | **PROGRAM SPECIFICATION AND EXECUTION** | 11 |  |
|  | 4.1 Screenshots | 15 |  |
| **5.** | **CONCLUSION** | 23 |  |
| **6.** | **REFERENCE** | 24 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**1.INTRODUCTION**

**1.1 HEART DISEASE PREDICTION:**

In the contemporary landscape of healthcare, the ability to predict and prevent heart disease has become a paramount focus. Heart disease, encompassing conditions such as coronary artery disease, heart failure, and arrhythmias, remains a leading cause of morbidity and mortality globally. Identifying individuals at risk and implementing preventive measures are critical steps in reducing the burden of heart-related ailments.

Heart disease prediction involves the use of advanced technologies and analytical methods to assess an individual's likelihood of developing cardiovascular issues. This proactive approach not only aids in early detection but also empowers individuals and healthcare professionals to take primitive actions to mitigate risks and promote heart health.

The advent of data-driven technologies, including machine learning and artificial intelligence, has revolutionized the field of predictive analytics in healthcare. These technologies enable the analysis of vast datasets comprising medical records, lifestyle factors, genetic information, and other relevant parameters. By discerning patterns and correlations within this data, predictive models can be developed to estimate an individual's susceptibility to heart disease.

One prevalent approach in heart disease prediction involves the integration of various risk factors. These factors may include age, gender, family history, blood pressure, cholesterol levels, diabetes status, and lifestyle choices such as diet and physical activity. Machine learning algorithms then process and learn from historical data to make accurate predictions regarding the likelihood of developing heart conditions.

Furthermore, hybrid models that combine traditional risk factor assessment with cutting-edge machine learning techniques are gaining prominence. These hybrid models leverage the strengths of both approaches, enhancing the accuracy and reliability of predictions. As a result, healthcare providers can offer more personalized recommendations and interventions tailored to an individual's unique risk profile.

The importance of heart disease prediction extends beyond individual health management. Public health initiatives can benefit significantly from these predictive models by identifying high-risk populations and implementing targeted interventions. By preventing the onset or progression of heart disease, these predictions contribute to a more cost-effective and sustainable healthcare system.

In this era of precision medicine, heart disease prediction emerges as a pivotal tool in the pursuit of proactive and personalized healthcare. By harnessing the capabilities of advanced technologies, healthcare professionals can move towards a future where heart disease is not just treated but anticipated and prevented, ultimately promoting a healthier and more resilient global population.

**1.2 PROJECT OBJECTIVE:**

In this fast-paced world, maintaining optimal health is a necessity for each individual to sustain vitality and well-being. Predicting and preventing heart disease plays a crucial role in rejuvenating our physical and emotional state. By proactively assessing our risk of heart-related issues, we empower ourselves to make informed lifestyle choices. Much like the convenience of movie recommendation systems for selecting preferred entertainment, the use of advanced technologies in heart disease prediction becomes essential to efficiently navigate through the vast landscape of health data.

Predicting heart disease provides a vital tool for individuals and healthcare professionals to tailor preventive strategies and interventions. Instead of spending valuable time searching for relevant health information, a heart disease prediction system proves to be a reliable solution. By leveraging predictive analytics, these systems analyze various risk factors such as medical history, lifestyle choices, and genetic predispositions, offering personalized insights into potential heart health concerns. In summary, much like the importance of entertainment in our lives, the significance of heart disease prediction cannot be overstated in the quest for a healthier and more balanced lifestyle. Integrating predictive technologies streamlines the process of understanding and addressing potential heart health risks, allowing individuals to focus on their well-being with the same efficiency and convenience as choosing their preferred form of entertainment.

**1.3 PROJECT SPECIFICATION:**

This project focuses on the development of an advanced Heart Disease Prediction System, offering users the ability to proactively manage their cardiovascular health. The system aims to streamline the health risk assessment process, providing an accessible and user-friendly platform for predicting the likelihood of heart-related issues.

**2.SYSTEM SPECIFICATION**

**2.1 HARDWARE SPECIFICATION:**

* Processor : Intel dual core
* Processor speed : 1.04GHZ
* Ram : 1GB
* Moniter
* Keyboard
* Mouse

**2.2** **SOFTWARE SPECIFICATION:**

* OS
* Language : Python
* Compiler : googlecolab

**3.PACKAGES**

* 1. **NUMPY:**
* NumPy is a Python library used for working with arrays.
* It also has functions for working in domain of linear algebra, fourier transform, and matrices.
* NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.
* NumPy stands for Numerical Python.

**INSTALLING NUMPY PACKAGE:**

pip install numpy

## WHY USE NUMPY?

* In Python we have lists that serve the purpose of arrays, but they are slow to process.
* NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.
* The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.
* Arrays are very frequently used in data science, where speed and resources are very important.

**IMPORT NUMPY**

Once NumPy is installed, import it in your applications by adding the import keyword:

import numpy

## NumPy as np:

Create an np with the as keyword while importing:

import numpy as np

Now the NumPy package can be referred to as np instead of numpy.

**Example:**

import numpy as np

arr = np.array([1, 2, 3, 4, 5])

print(arr)

## 0-D Arrays

0-D arrays, or Scalars, are the elements in an array. Each value in an array is a 0-D array.

## 1-D Arrays

An array that has 0-D arrays as its elements is called uni-dimensional or 1-D array.

These are the most common and basic arrays.

## 2-D Arrays

An array that has 1-D arrays as its elements is called a 2-D array.

These are often used to represent matrix or 2nd order tensors.

**3.2 PANDAS:**

* Pandas is a Python library used for working with data sets.
* It has functions for analyzing, cleaning, exploring, and manipulating data.
* The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.

## WHY USE PANDAS?

* Pandas allows us to analyze big data and make conclusions based on statistical theories.
* Pandas can clean messy data sets, and make them readable and relevant.
* Relevant data is very important in data science.

Pandas gives you answers about the data:

* Is there a correlation between two or more columns?
* What is average value?
* Max value?
* Min value?
* Pandas are also able to delete rows that are not relevant, or contains wrong values, like empty or NULL values. This is called cleaning the data.

**INSTALLING PANDAS PACKAGE**

pip install pandas

## IMPORT PANDAS

Once Pandas is installed, import it in your applications by adding the import keyword:

import pandas

Now Pandas is imported and ready to use

**Example:**

import pandas as pd

mydataset={'cars':["BMW","Volvo","Ford"],'passings':[3,7,2]}  
myvar=pandas.DataFrame(mydataset)  
print(myvar)

## Pandas as pd

Pandas is usually imported under the pd

Create an pd with the as keyword while importing:

import pandas as pd

Now the Pandas package can be referred to as pd instead of pandas.

**3.2.1 PANDAS DATAFRAME:**

A DataFrame is a two-dimensional, tabular data structure with labeled axes (rows and columns).

• It is similar to a spreadsheet or a SQL table and is capable of handling various data types.

• The rows and columns in a Pandas DataFrame can be labeled, making it easy to perform operations on the data.

FUNCTIONS OF PANDAS DATAFRAME:

• Merge - Merges two DataFrames based on a specified column.

• concat() - Concatenates DataFrames along a particular axis.

• plot() - Generates various types of plots using Matplotlib.

• heatmap() - Creates a heatmap for visualizing data.

• head() – Generates first five lines of your data set.

• tail() – Generates last five lines of your data set.

**3.3 MATPLOTLIB:**

* Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy.
* As such, it offers a viable open source alternative to **MATLAB.** Developers can also use matplotlib’s APIs(Application Programming Interfaces) to embed plots inGUI applications.

A Python matplotlib script is structured so that a fewlines of code are all that is required in most instancesto generate a visual data plot.

The matplotlib scripting layer overlays two APIs:

* The pyplot API is a hierarchy of Python codeobjects topped by matplotlib.pyplot
* An OO (Object-Oriented) API collection of objectsthat can be assembled with greater flexibility thanpyplot. This API provides direct access to Matplotlib’sbackend layers.

**Matplotlib and Pyplot in Python:**

The pyplot API has a convenient MATLAB-style statefulinterface. In fact, matplotlib was originally written as an open source alternative for MATLAB. The OO API and its interface is more customizable and powerful than pyplot, but considered more difficult to use. As a result, the pyplot interface is more commonly used, and is referred to by default in this article.

Understanding matplotlib’s pyplot API is key to understanding how to work with plots:

* **matplotlib.pyplot.figure**: Figure is the top-level container. It includes everything visualized in a plot including one or more Axes.
* **matplotlib.pyplot.axes**: Axes contain most of the elements in a plot: Axis, Tick, Line2D, Text, etc., and sets the coordinates. It is the area in which data is plotted. Axes include the X-Axis, Y-Axis, and possibly a Z-Axis, as well.

**INSTALLING MATPLOTLIB:**

pip install matplotlib

**3.3.1 MATPLOTLIB BAR PLOT:**

A bar plot or bar chart is a graph that represents the category of data with rectangular bars with lengths and heights that is proportional to the values which they represent. The bar plots can be plotted horizontally or vertically. A bar chart describes the comparisons between the discrete categories. One of the axis of the plot represents the specific categories being compared, while the other axis represents the measured values corresponding to those categories.

**Creating a bar plot:**

The matplotlib API in Python provides the bar() function which can be used in MATLAB style use or as an object-oriented API. The syntax of the bar() function to be used with the axes is as follows:- plt.bar(x, height, width, bottom, align).The function creates a bar plot bounded with a rectangle depending on the given parameters. Following is a simple example of the bar plot, which represents the number of students enrolled in different courses of an institute.

**Example:**

**import numpy as np**

**import matplotlib.pyplot as plt**

**pd.crosstab(df.target,df.sex).plot(kind='bar',figsize=(10,6),color=['red','green'])**

**plt.title('Summary of Heart Disease sex wise')**

**plt.xlabel('0=No Heart disease,1=Heart Disease')**

**plt.ylabel('Number of Individual')**

**plt.legend(['Male','Female'])**

**plt.xticks(rotation=0)**

**plt.show()**

**Output:**

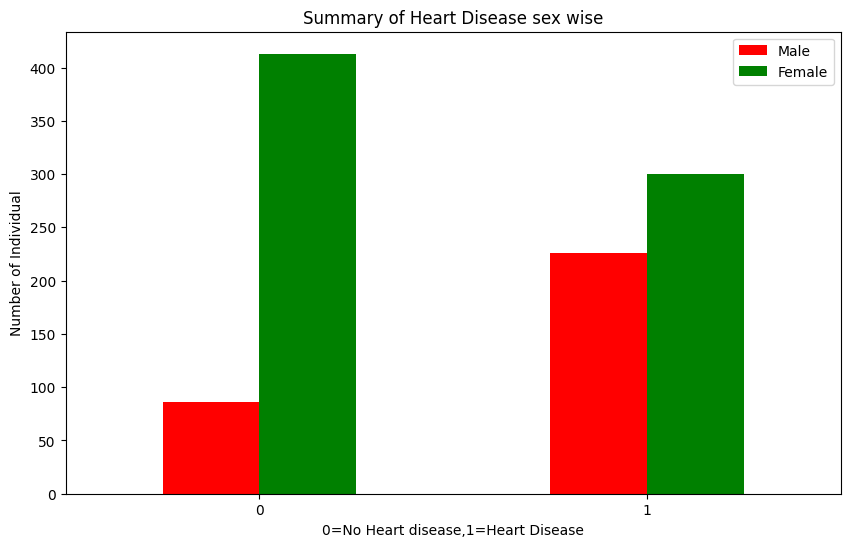


Figure 1: Bar Chart

**3.3.2 MATPLOTLIB HISTOGRAM:**

A histogram is an accurate representation of the distribution of numerical data. It is an estimate of the probability distribution of a continuous variable. It is a kind of bar graph.

To construct a histogram, follow these steps −

* Bin the range of values.
* Divide the entire range of values into a series of intervals.
* Count how many values fall into each interval.

The bins are usually specified as consecutive, non-overlapping intervals of a variable.

The **matplotlib.pyplot.hist()** function plots a histogram. It computes and draws the histogram of x.

**Example:**

from matplotlib import pyplot as plt

import numpy as np

df.age.plot.hist()

plt.show()

**Output:**

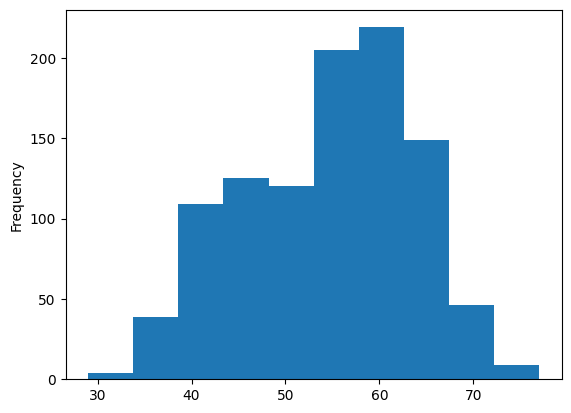


Figure 2: Histogram

**4.PROJECT DESCRIPTION AND EXECUTION**

pip install --upgrade scikit-learn

The command pip install --upgrade scikit-learn is used to upgrade the scikit-learn library to the latest version. This command should be run in your command line or terminal, depending on your operating system.

import pandas as pd   
import numpy as np  
import warnings  
warnings.filterwarnings('ignore')

Next, we load in the data set using pandas read\_csv() utility. The dataset is tab separated so we pass in \t to the sep parameter. We then pass in the column names using the names parameter.

df = pd.read\_csv('u.data', sep='\t', names=['user\_id','item\_id','rating','titmestamp'])

df=pd.read\_csv('/content/archive (1).zip')

Now let’s check the head of the data to see the data we are dealing with.

df.head()

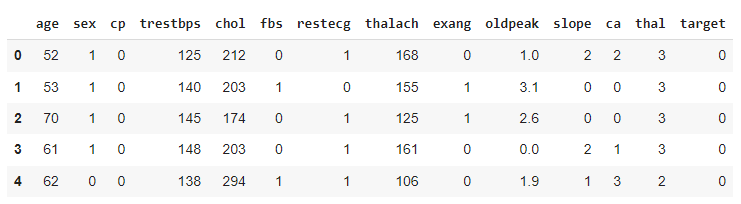


Figure 3: Head of the data set

Check the tail of the data to see the data we are dealing with.

df.tail()

To provides a summary of descriptive statistics for the numerical columns in a DataFrame. It is a powerful tool for quickly analyzing and understanding the distribution of data.

df.describe()

We can see that the dataset has 1025 records.

Let’s now create a data frame with the average rating for each heart rate and the number of ratings. We are going to use these ratings to calculate the correlation between the datas later. Correlation is a statistical measure that indicates the extent to which two or more variables fluctuate together. Ratings that have a high correlation coefficient are the ratings that are most similar to each other. In our case, we shall use the Pearson correlation coefficient. This number will lie between -1 and 1. 1 indicates a positive linear correlation while -1 indicates a negative correlation. 0 indicates no linear correlation. Therefore, movies with a zero correlation are not similar at all. In order to create this data frame we use pandas groupby functionality. We group the dataset by the title column and compute its mean to obtain the average rating for each data.

df.corr()

home,room=plt.subplots(figsize=(15,15))

room=sns.heatmap(df.corr(),annot=True,linewidths=0.5,fmt='0.2f',cmap='YlGnBu')

Next we would used to count the occurrences of unique values in the 'sex' column. This is particularly useful when you want to understand the distribution of values in a categorical column like 'sex'.

df.sex.value\_counts()

Here's a breakdown of what this code does:

df: Assuming df is your Pandas DataFrame.

sex: Refers to the column name 'sex' in the DataFrame.

value\_counts(): A Pandas function that returns a Series containing counts of unique values. In this case, it counts the occurrences of each unique value in the 'sex' column.

Let’s now plot a Histogram using pandas plotting functionality to visualize the frequency distribution of age.

import matplotlib.pyplot as plt

df.age.plot.hist()

Now let’s look at the logistic regression.

Logistic regression is a type of supervised learning algorithm used for binary classification tasks in machine learning. It's called "logistic" regression because it uses the logistic function to model a binary dependent variable. Despite its name, logistic regression is used for classification, not regression.

To use this we are going to import some modules.

from sklearn.linear\_model import LogisticRegression

from sklearn.model\_selection import train\_test\_split

In scikit-learn's LogisticRegression, the max\_iter parameter determines the maximum number of iterations taken for the solver to converge. The solver is the optimization algorithm used to fit the logistic regression model to the training data. If the solver doesn't converge within the specified number of iterations, it stops and may not have found the optimal parameters.

max\_iter: This parameter represents the maximum number of iterations for the solver to converge. The solver is an optimization algorithm used to find the optimal parameters (weights) that minimize the logistic regression cost function.

model=LogisticRegression(max\_iter=1000)

Next to split your dataset into training and testing sets. This is a common practice in machine learning to assess the performance of a model on data it has not seen during training.

Here's what each part of the code does:

x: This represents your feature matrix or input data.

y: This represents your target variable or output labels.

The function train\_test\_split is part of the scikit-learn library and is used to split the dataset into training and testing sets. Here's what each parameter does:

x\_train: This will contain the features used for training the model.

x\_test: This will contain the features used for testing the model.

y\_train: This will contain the target values corresponding to x\_train.

y\_test: This will contain the target values corresponding to x\_test.

test\_size: This is the proportion of the dataset to include in the test split. In this case, it's set to 0.3, meaning 30% of the data will be used for testing, and 70% for training.

random\_state: This is an arbitrary number that, when specified, ensures reproducibility. If you use the same random\_state, you'll get the same split each time you run your code. This is useful for reproducibility and debugging.

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.3,random\_state=0)

To train your machine learning model using the training data. Here's a breakdown of what this line of code does:

model: This represents the machine learning model you've instantiated. It could be a classifier or a regressor, depending on the nature of your task (classification or regression).

fit(): This is a method in scikit-learn used to train the model.

x\_train: This is your training data, typically the feature matrix. It contains the input features used to train the model.

y\_train: This is the corresponding target variable or labels for the training data. It contains the correct output values that the model will try to predict.

model.fit(x\_train,y\_train)

Score method in scikit-learn is used to evaluate the accuracy of a trained model on the training data. It returns the coefficient of determination R2 of the prediction.

model.score(x\_train,y\_train)

In this method, it explains:

model: Represents your trained machine learning model.

x\_test: Represents the feature matrix of your testing data.

y\_test: Represents the corresponding target variable or labels for the testing data.

The model.score() method internally makes predictions using the trained model on the input data (x\_test) and compares these predictions with the actual labels (y\_test). It then returns a measure of accuracy or goodness of fit on the test set.

model.score(x\_test,y\_test)

To train data and calculates their respective training and testing scores.

train\_score = []

test\_score = []

logistic\_params = range(1, 30)

for i in logistic\_params:

LR = LogisticRegression(max\_iter=5000, C=i)

LR.fit(x\_train, y\_train)

train\_score.append(LR.score(x\_train, y\_train))

test\_score.append(LR.score(x\_test, y\_test))

To visualize the training and testing scores across different values of the logistic regression parameter using the matplotlib library.

plt.figure(figsize=(15, 8))

plt.plot(logistic\_params, train\_score, label='Train Score')

plt.plot(logistic\_params, test\_score, label='Test Score')

plt.xticks(np.arange(1, 31, 1))

plt.xlabel('Logistic Regression Parameter (C)')

plt.ylabel('Model Accuracy')

plt.legend()

plt.show()

print(f'Max Logistic Regression Score: {max(test\_score)\*100:0.2f}%')

Next, We use Random Forest algorithm and Naïve Bayes algorithm to see what algorithm provides high accuracy rate which was similar to logistic regression algorithm.

The accuracy rate was,

* Logistics Regression - 0.8668831168831169
* Random Forest - 0.9609758070657922
* Naïve Bayes - 0.8441558441558441

From the above accuracies,

* Random Forest achieved the highest accuracy among the three models (96.10%), suggesting that, based on the evaluation metric used, it performed the best on the given heart disease prediction task.
* Logistic Regression also performed well with an accuracy of 86.69%.
* Naïve Bayes had the lowest accuracy among the three models (84.42%).

However, accuracy is just one metric, and it might not be sufficient for a comprehensive evaluation, especially in imbalanced datasets. Depending on the characteristics of your data and the specific requirements of your task, you might also want to consider other metrics such as precision, recall, F1 score, or the area under the ROC curve (AUC-ROC) to assess the models' performance more thoroughly.

**4.1 SCREENSHOT:**

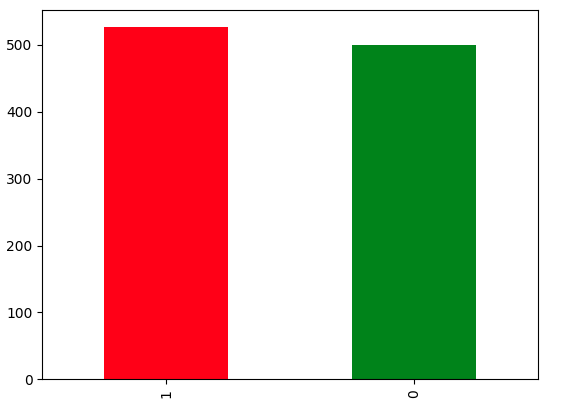


Figure 4: Bar chart

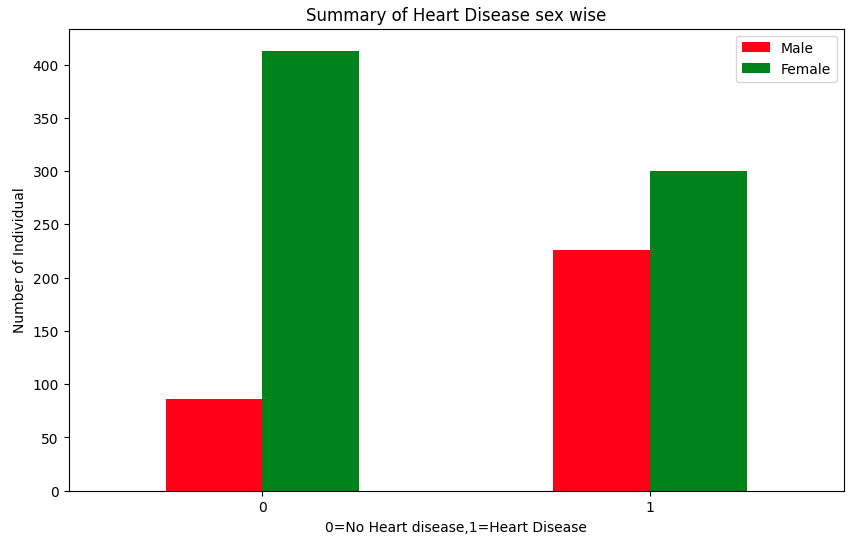


Figure 5: Heart disease sex wise

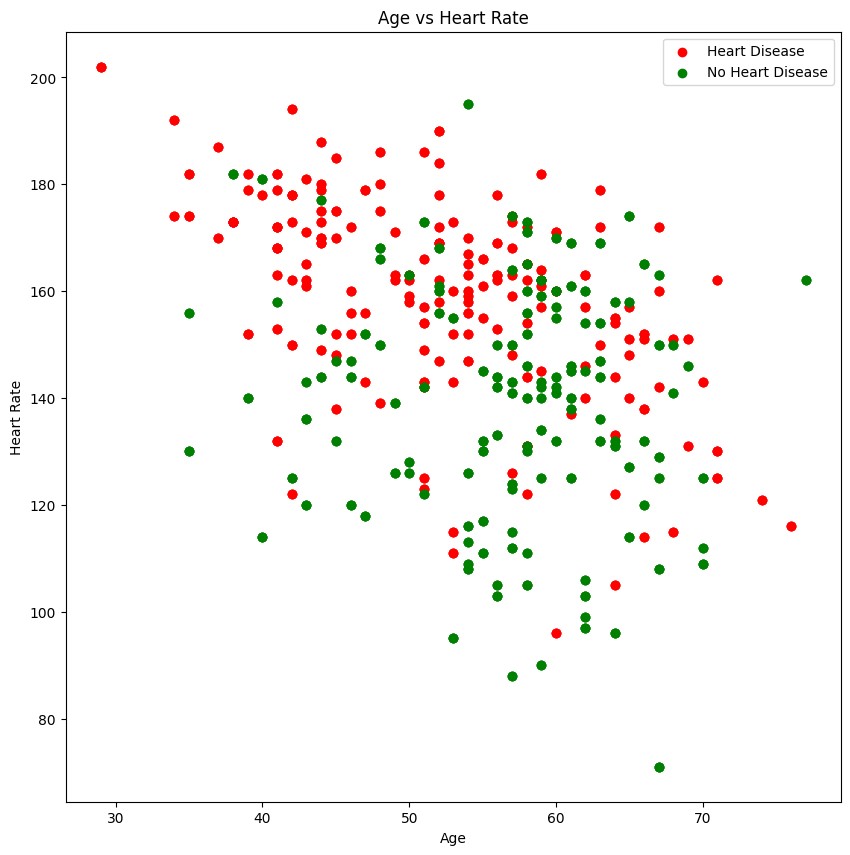


Figure 6: Scatter Plot

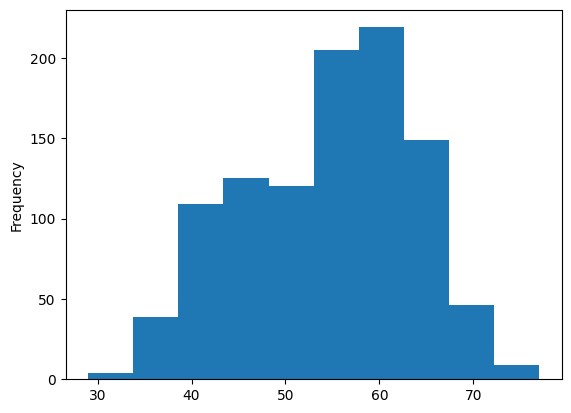
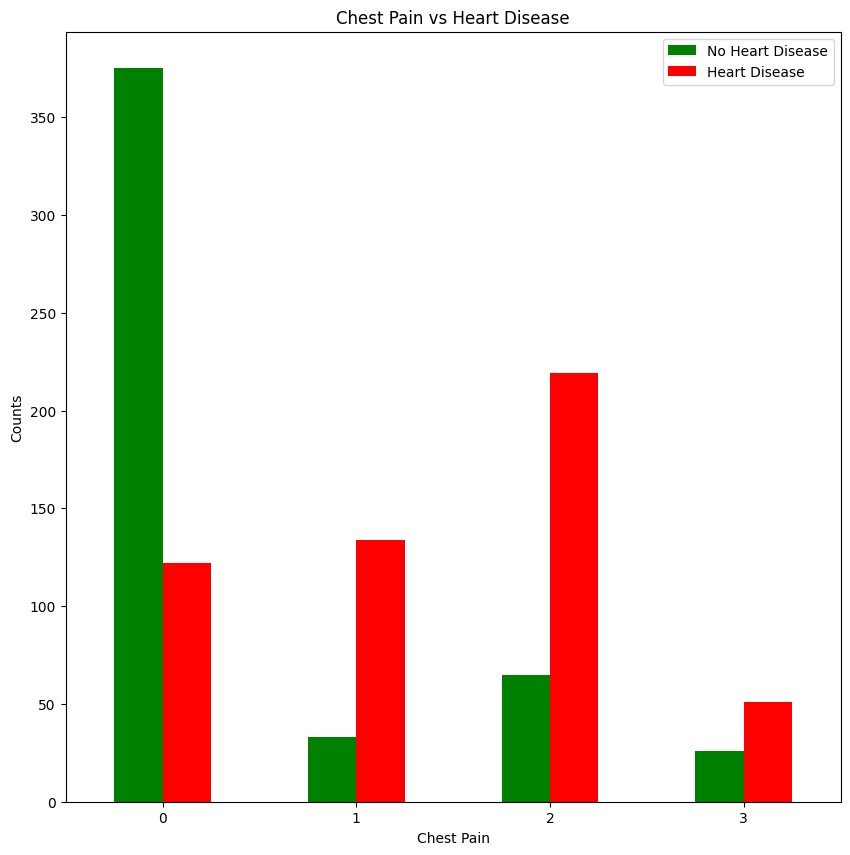


Figure 7: Histogram showing age distribution

Figure 8: Bar chart

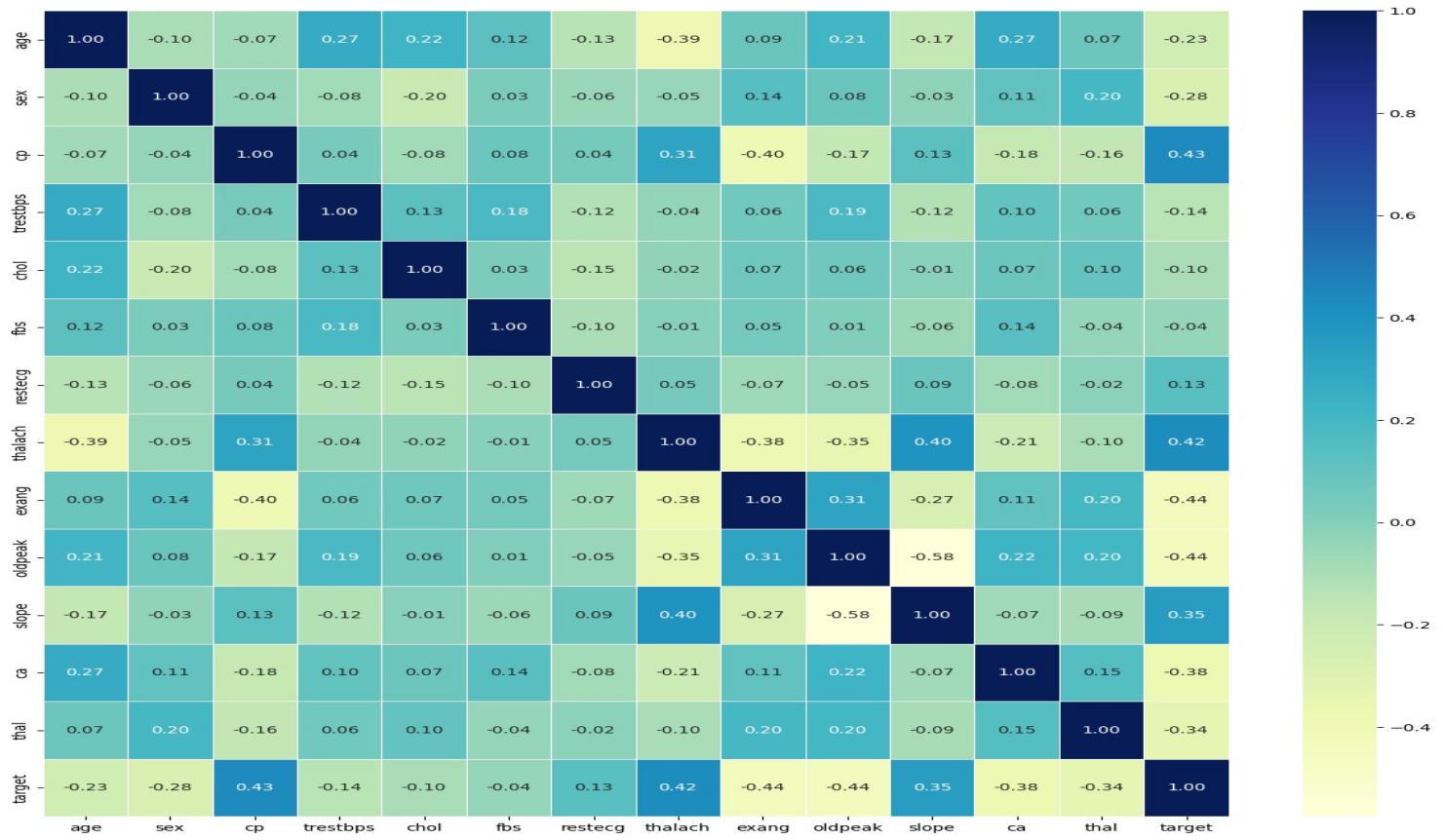


Figure 9: Correlation Matrix

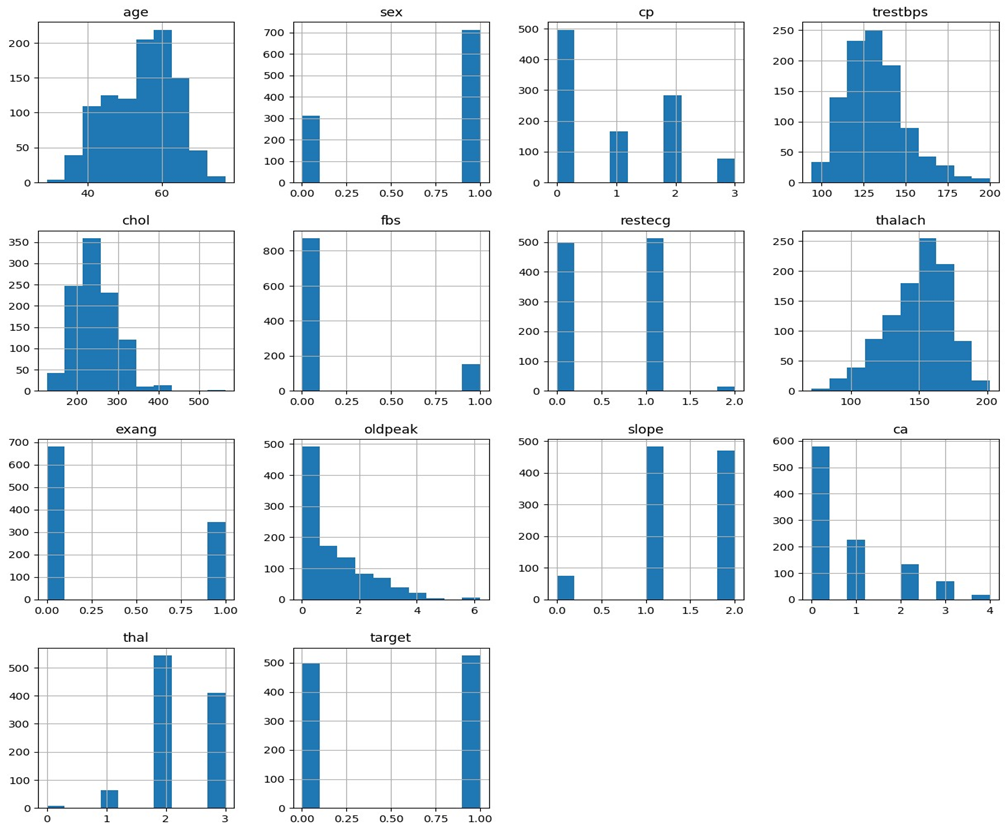


Figure 10: Histogram Grid

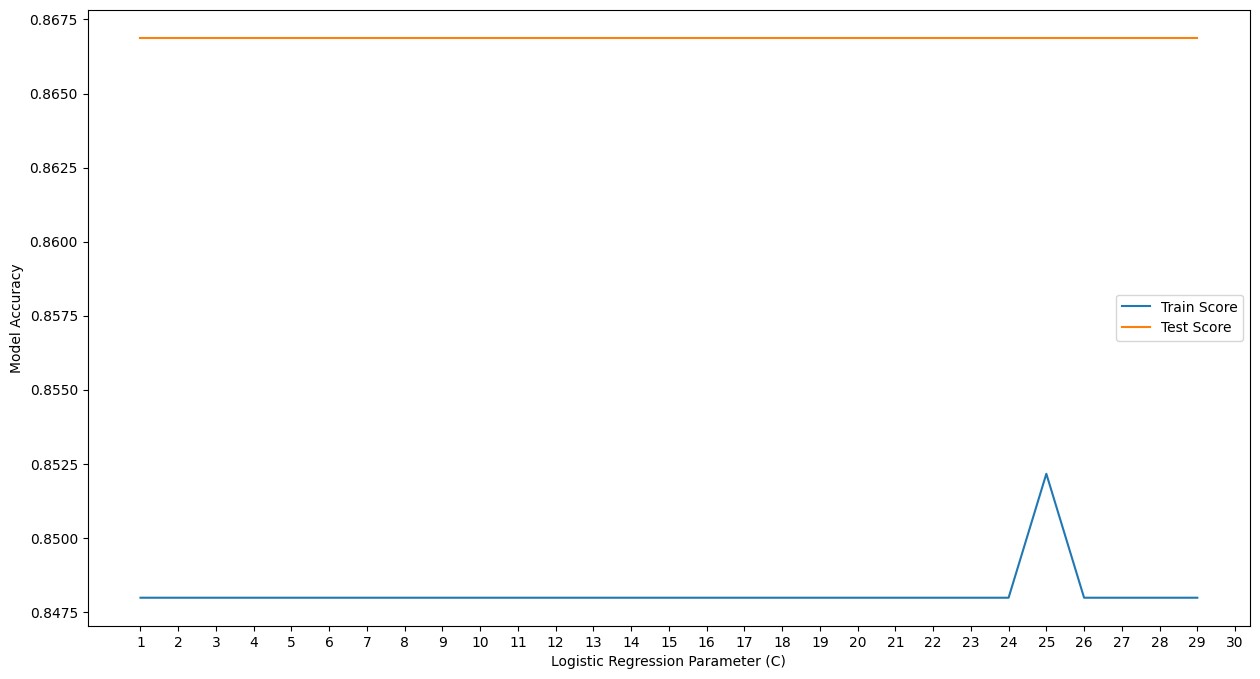


Figure 11: Logistic Regression Parameter(C)

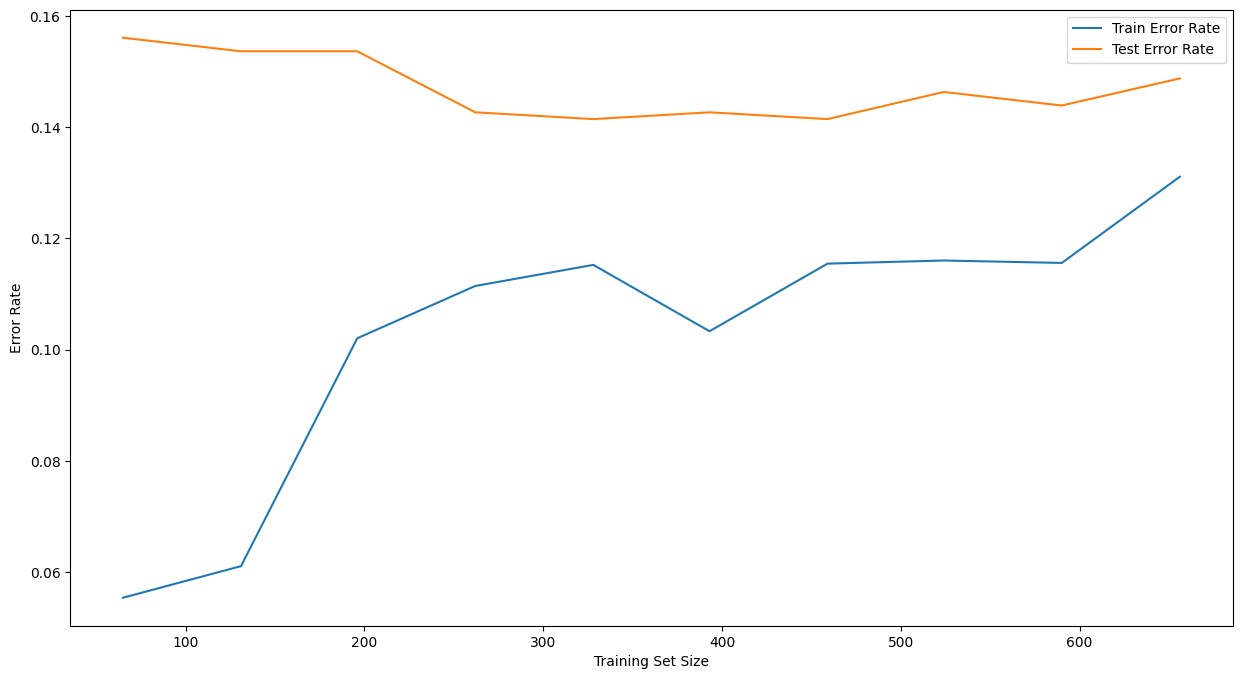


Figure 12:Logistic Regression(Error Rate)

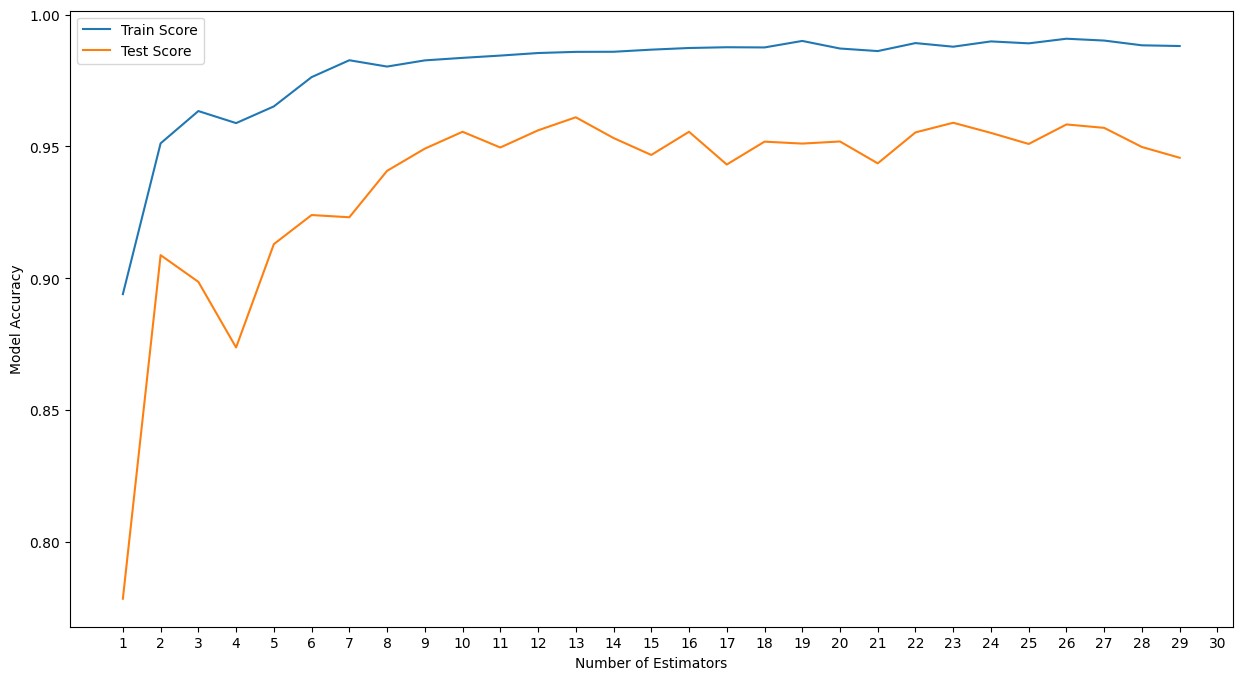


Figure 13:Random Forest Regressor Performance

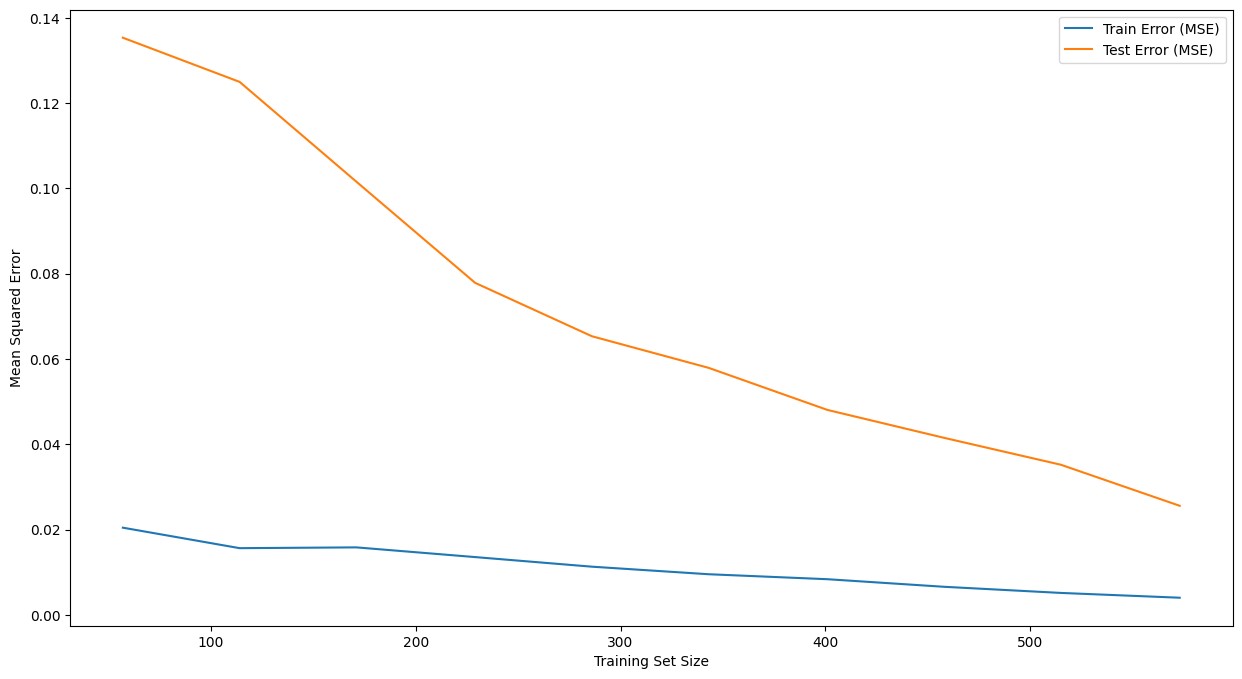


Figure 14:Random Forest(Error Rate)

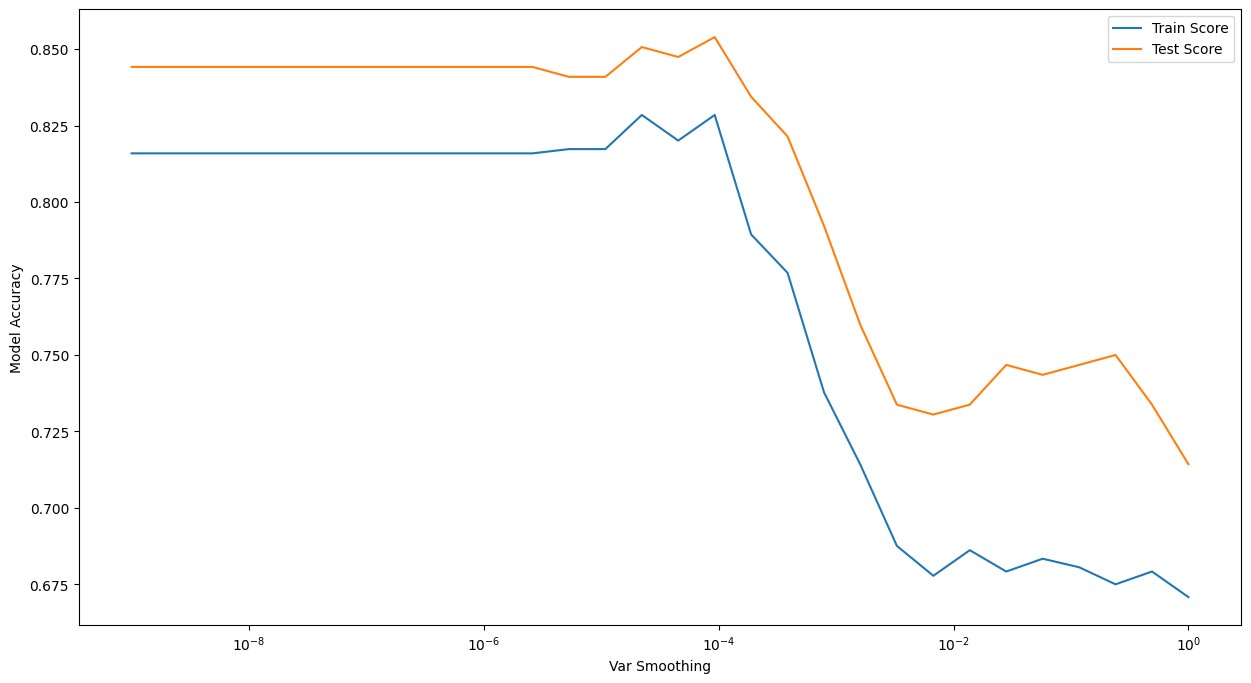


Figure 15:Naive Bayes Performance

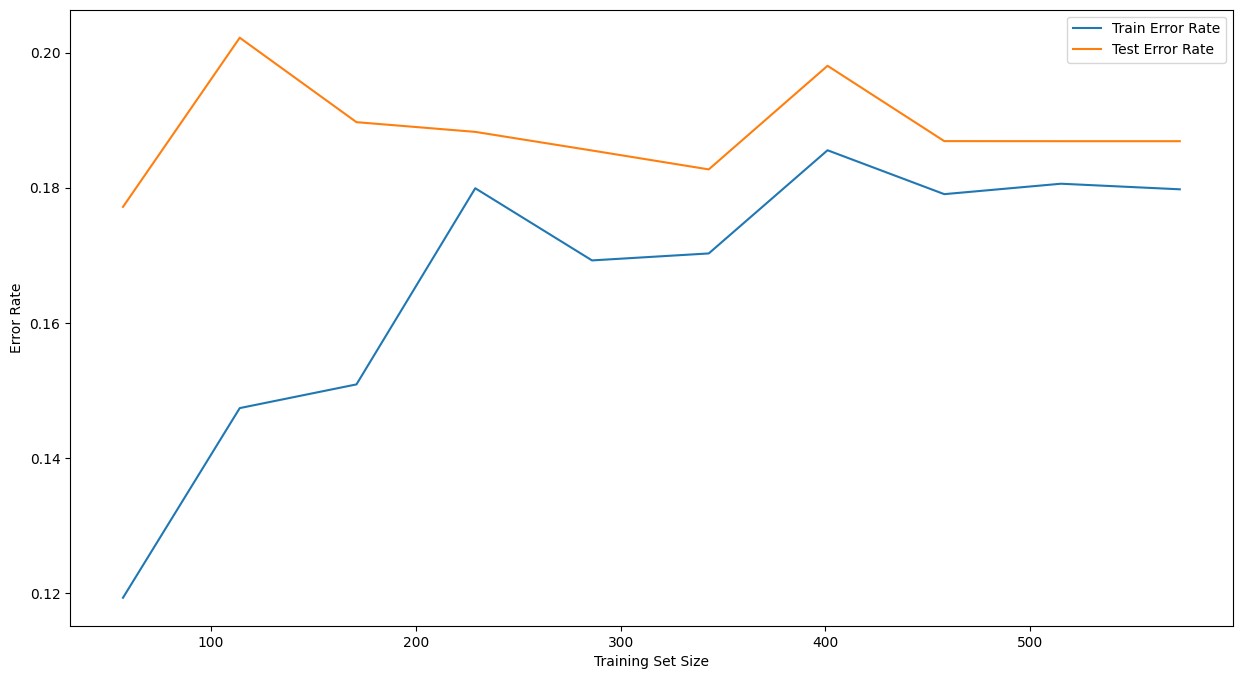


Figure 16:Naive Bayes(Error Rate)

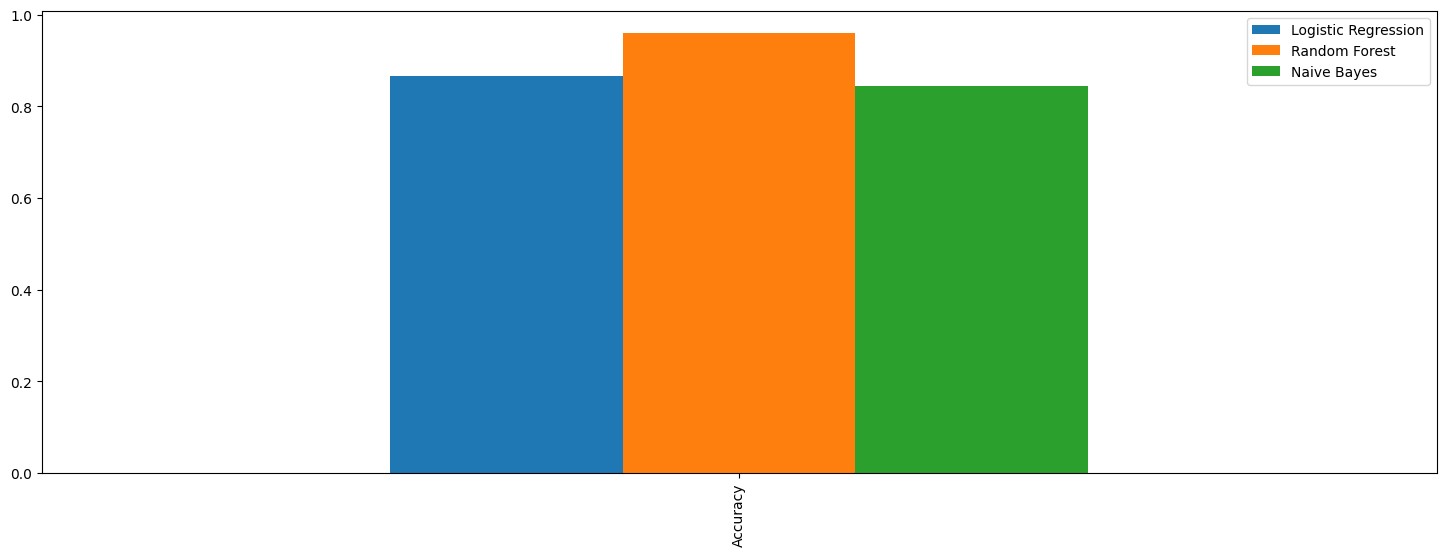


Figure 17:Accuracy(Bar Chart)

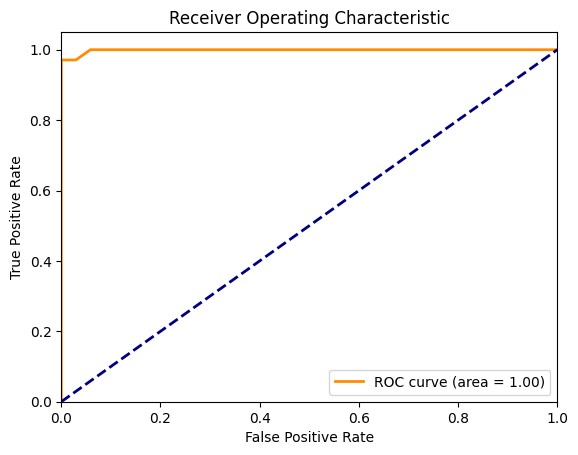


Figure 18:Random Forest(ROC Curve)

**5. CONCLUSION**

In conclusion, the development of the Heart Disease Prediction System represents a significant stride toward proactive and personalized healthcare. The system, designed to assess an individual's risk of developing heart-related conditions, brings accessibility and convenience to the forefront of health management. With the integration of predictive modeling and machine learning algorithms, users can now easily access personalized insights into their cardiovascular health from anywhere, fostering a more informed and engaged approach to well-being.

This initiative not only streamlines the health risk assessment process but also empowers individuals to take charge of their heart health through early intervention and preventive measures. The user-friendly interface ensures that health predictions are readily comprehensible, encouraging a deeper understanding of individual risk factors and promoting healthier lifestyle choices.

By incorporating educational resources and personalized recommendations, the Heart Disease Prediction System extends beyond a mere diagnostic tool, serving as a holistic platform for health awareness and continuous improvement. The integration with health records further enhances the accuracy of predictions, fostering collaboration between users and healthcare professionals.

In essence, this project strives to contribute to a healthier society by leveraging advanced technologies to predict and prevent heart disease. The anywhere, anytime accessibility of health insights aims to revolutionize the traditional approach to healthcare, placing the power of personalized health management firmly in the hands of individuals. Through the deployment of stringent privacy measures and adherence to ethical guidelines, this Heart Disease Prediction System seeks to redefine how we approach health, ushering in an era of proactive, data-driven, and user-centric healthcare solutions.

**6.REFERENCE**

**WEBSITES:**

1. [https://machinelearningmastery.com/overfitting-machine-learning-models/](https://machinelearningmastery.com/overfitting-machine-learning-models/#inbox/_blank)
2. [https://stackoverflow.com/questions/43979449/higher-validation-accuracy-than-training-accurracy-using-tensorflow-and-keras](https://stackoverflow.com/questions/43979449/higher-validation-accuracy-than-training-accurracy-using-tensorflow-and-keras#inbox/_blank)
3. [https://machinelearningmastery.com/how-to-calculate-precision-recall-f1-and-more-for-deep-learning-models/](https://machinelearningmastery.com/how-to-calculate-precision-recall-f1-and-more-for-deep-learning-models/#inbox/_blank)

|  |  |
| --- | --- |
| **PERFORMANCE** |  |
| **VIVAVOCE** |  |
| **MINI PROJECT** |  |
| **TOTAL** |  |