A project report on

Heart Disease Prediction using Machine Learning

By

Satya Uma Praneetha V Parupudi, Balaswamy Rusum and Sai Mokshith Reddy Peta

Course: Data Mining CSCI-57300

INTRODUCTION

Heart disease, specifically cardiovascular disease (CVDs), is a major global health concern, responsible for more than 70% of worldwide deaths. The prevalence of CVDs is not limited to high-income nations, as low- and middle-income countries also experience a rise in chronic illness. The economic burden of cardiovascular diseases is substantial, reaching approximately USD 3.7 trillion between 2010 and 2015. The excessive cost and impracticality of diagnostic technologies contribute to delayed detection, resulting in significant human and financial costs. To address this, leveraging data mining and machine learning methods for early detection becomes crucial.

1.1 PROBLEM STATEMENT

The challenge lies in the efficient and timely detection of heart disease, given its global impact on morbidity and mortality. Current diagnostic technologies are often expensive and impractical for widespread use, leading to many preventable deaths. The economic burden on individuals and organizations is substantial, with cardiovascular diseases accounting for a massive portion of annual medical expenses.

The World Health Organization predicts a further increase in CVD-related deaths, emphasizing the need for proactive measures. To address these challenges, this research focuses on the application of data mining and machine learning techniques to predict the likelihood of developing heart disease. The goal is to explore the effectiveness of various machine learning algorithms using a diverse dataset, aiming for more accurate predictions and improved generalizability compared to previous studies with smaller sample sizes.

In the medical field, data mining reveals hidden patterns for clinical diagnosis. Diabetes, high blood pressure, high cholesterol, and abnormal pulse rate are

essential factors to consider when predicting heart disease. However, incomplete medical data can affect prediction accuracy.

Machine learning plays a pivotal role in diagnosing, detecting, and predicting diseases. Current research focuses on using machine learning techniques, such as random forest, decision tree classifier, multilayer perceptron, to predict heart disease. The study aims to assess the effectiveness of these algorithms using a diverse dataset, addressing previous limitations in sample size and generalizability. The goal is to improve the accuracy of predicting heart disease, contributing to saving lives and reducing the economic burden on society.

1.2 OVERVIEW OF THE DATA

Dataset source: This dataset was created by combining five independent heart disease datasets, resulting in a comprehensive dataset for research purposes. The datasets used for curation are:

- Cleveland: 303 observations
- Hungarian: 294 observations
- Switzerland: 123 observations
- Long Beach VA: 200 observations
- Stalog (Heart) Data Set: 270 observations

Total: 1190 observations
Duplicated: 272 observations
Final dataset: 918 observations with 11 common features

Attributes:

SL. no	Attribute	Description	dType
1	Age	Age is in range 29-77 years.	int64
2	Sex	Male: 0 Female: 1	int64
3	Chest Pain	Different types of Chest pain, ATA, NAP, ASY, TA	int64
4	Resting BP	Resting Blood pressure in mm HG range: 94-200	int64
5	Cholesterol	Serum Cholesterol in mg/del, range: 126-564	int64
6	Fasting BS	Range < and > 120mg/dl True: 1 False:0	int64
7	Resting ECG	Resting Electrocardiogram Normal, ST, LVH	int64
8	Max HR	Max heart rate 71-202	int64
9	Exercise Agnima	Pain during exercise Yes: 1 or No :0	int64
10	Old peak	ST depression due exercise w.r.t rest 0 to 2	int64
11	ST Slope	Slope of peak exercise, Up, Flat, down	int64
12	Heart disease	Class label. Cardiac Disease yes: 1, no: 0	int64

1.3 Team Contribution

Each member of the team played a significant role in developing a robust machine learning model for heart disease prediction with the given clinical data.

Sai Mokshith Reddy Peta played a pivotal role in the initial stages of our research, focusing on data pre-processing and data visualization. His expertise in handling and transforming raw data into a suitable format for machine learning models significantly contributed to the overall quality of our dataset. Through meticulous analysis and visualization techniques, he provided valuable insights that laid the foundation for subsequent stages of our research. With his help, the Random Forest and Logistic regression models were built. He helped in building model architectures, carrying out literature surveys.

Satya Uma Praneetha V Parupudi led the literature survey, Exploratory Data Analysis (EDA) phase of our study and the future direction for this project. Her keen analytical skills and in-depth exploration of the dataset unearthed hidden patterns and relationships among variables. Furthermore, Satya Uma Praneetha V Parupudi actively engaged in feature scaling, one-hot encoding, and label encoding of features. Explored the dimensionality reduction techniques (PCA) to attain better accuracy and concluded that, as the data is too small, PCA would increase variance and information loss. Built ML algorithms, for Decision tree, Support vector Machine. Calculated the performance metrics, and the ROC curves, and compared the performance metrics of different models for applied grid search data.

Balaswamy Rusum took charge of the stages involving building Naive Bayes, multi-layer perceptron and model selection, Grid Search Cross-Validation (CV), hyperparameter tuning, and model evaluation. His expertise in selecting appropriate machine learning models, fine-tuning hyperparameters, and conducting extensive model evaluations on various metrics, with a primary focus on recall, significantly shaped the predictive performance of our heart disease prediction model. Balaswamy Rusum dedication to optimizing model performance underscored the reliability and effectiveness of our proposed methodology.

2 RELATED WORK

Till date different studies have been done on heart disease prediction. Various data mining and machine learning algorithms have been implemented and proposed on the datasets of heart patients and different results have been achieved for different techniques. But still today we are facing a lot of problems caused by heart disease. Some of the recent research papers are as follows,

1 In 2018, Navdeep Singh and Sonika Jindal devise a model called Hybrid Genetic Naive Bayes Model using two supervised machine learning algorithms i.e., Genetic algorithm and Naive Bayes to predict heart disease with high accuracy results. In this propped model a dataset of 303 records with 14 necessary attributes has been taken from the online Cleveland database of UCI repository. The results are obtained in the type of the various performance parameters as precision (precision=98 %), recall (recall=97.14%) and accuracy (accuracy=97.14%).

2 In a study published by Alotalibi (2019), the author aimed to investigate the utility of machine learning (ML) techniques for predicting heart failure disease. The study utilized a dataset from the Cleveland Clinic Foundation, and implemented various ML algorithms, such as decision tree, logistic regression, random forest, naive Bayes, and support vector machine (SVM), to develop prediction models. A 10-fold cross-validation approach was employed during the model development process. The results indicated that the decision tree algorithm achieved the highest accuracy in predicting heart disease, with a rate of 93.19%, followed by the SVM algorithm at 92.30%. This study provides insight into the potential of ML techniques as an effective tool for predicting heart failure disease and highlights the decision tree algorithm as a potential option for future research.

3 Through a comparison of multiple algorithms, Hasan, and Bao (2020) carried out a study with the main objective of identifying the most efficient feature selection approach for anticipating cardiovascular illness. The three well-known feature selection methods (filter, wrapper, and embedding) were first considered, and then a feature subset was recovered from these three algorithms using a Boolean process-based common “True” condition. This technique involved retrieving feature subsets in two stages. Several models, including random forest, support vector classifier, k-nearest neighbors, naive Bayes, and XGBoost, were considered to justify the comparative accuracy and identify the best predictive analytics. As a standard for comparison with all

features, the artificial neural network (ANN) was used. The findings demonstrated that the Algorithms 2023, 16, 88 4 of 14 most accurate prediction results for cardiovascular illness were provided by the XGBoost classifier coupled with the wrapper technique. XGBoost delivered an accuracy of 73.74%, followed by SVC with 73.18% and ANN with 73.20%.

4 In a study conducted by Shah et al. (2020) [18], the authors aimed to develop a model for predicting cardiovascular disease using machine learning techniques. The data used for this purpose were obtained from the Cleveland heart disease dataset, which consisted of 303 instances and 17 attributes, and was sourced from the UCI machine learning repository. The authors employed a variety of supervised classification methods, including naive Bayes, decision tree, random forest, and k-nearest neighbor (KKN). The results of the study indicated that the KKN model exhibited the highest level of accuracy, at 90.8%. The study highlights the potential utility of machine learning techniques in predicting cardiovascular disease and emphasizes the importance of selecting appropriate models and techniques to achieve optimal results.

5 Comparative Study on Heart Disease Prediction Using Feature Selection Techniques on Classification Algorithms. (2021) the authors, used The Cleveland heart disease dataset gathered from the UCI ML Repository contains 303 samples: 164 samples indicate the absence of heart disease, and 139 samples indicate the presence of heart disease. The optimal feature subsets for classification were selected by using ten feature selection algorithms, namely, Chi-square, ANOVA, mutual information, Relief, forward feature selection (FFS), backward feature selection (BFS), exhaustive feature selection (EFS), recursive feature elimination (RFE), Lasso (L1) regression, and Ridge (L2) regression categorized under the filter, wrapper, and embedded methods. The performance is evaluated using six classifiers: random forest, support vector machine, decision tree, K- nearest neighbor, logistic regression, and Gaussian Naive Bayes.

3 METHODOLOGY

Our project aimed to predict the probability of heart disease through algorithms which can be beneficial for medical professionals and patients. To achieve this, we

have used the various machine learning algorithms on the data set and are presenting the results in this project report.

3.1 Data Collection

Our data, from the Kaggle source has a shape of 918*12 I.e., total of 918 observations and 12 attributes. Out of the 12 attributes there are categorical and numerical attributes of 7,5. In data analysis and machine learning tasks, understanding the nature of features is crucial for choosing appropriate statistical methods or machine learning algorithms. Different types of features may require different preprocessing techniques and handling strategies during model development. Numerical Features represent measurable quantities and are expressed with numerical values. Examples: Age, Resting BP, Cholesterol, Max HR, Old peak. Operations like addition, subtraction, multiplication, and division are meaningful for numerical features. They can be continuous or discrete. Categorical features represent categories or labels and take on values from a finite set of distinct categories. Examples: Gender, chest pain type, Fasting BS, Resting ECG, Exercise Angina, ST Slope, and the class label heart disease. Operations like counting occurrences, mode, and frequency are meaningful for categorical features. They can be nominal (unordered categories) or ordinal (ordered categories).

3.2 Data Pre-processing

Data preprocessing is a crucial step in the machine learning pipeline. It involves cleaning and transforming raw data into a format that is suitable for analysis or training machine learning models. Let us discuss the importance of two specific data preprocessing steps,

Null and Duplicates: Identifying and handling null values (missing data) ensures the overall quality of the dataset. Null values can lead to biased or inaccurate analyses or machine learning models. Detecting and removing duplicate entries maintains data integrity. Duplicate records might skew statistical analyses or machine learning model training, leading to unreliable results.

Feature Scaling on Numerical Features: Many machine learning algorithms are sensitive to the scale of numerical features. Scaling ensures that no single feature dominates the others, preventing bias in the learning process. Scaling can improve

the convergence speed of optimization algorithms, particularly for methods like gradient descent used in training many machine learning models.

Algorithms that rely on distances between data points, such as support vector machines, benefit from feature scaling as it ensures that all features contribute equally to the distance computation. Methods used are,

Standardization: It involves transforming data to have a mean of 0 and a standard deviation of 1. This is done by subtracting the mean and dividing by the standard deviation. Here is the formula, for Standardization,

$$X' = \frac{X - \mu}{\sigma}$$

Normalization (Min-Max Scaling): It scales the data to a specific range (e.g., [0, 1]) by subtracting the minimum value and dividing it by the range (difference between maximum and minimum values). Here is the formula for Normalization,

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

3.3 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is the process of analyzing and visualizing data sets to summarize their main characteristics, often with the help of statistical graphics and other data visualization methods. EDA helps uncover patterns, relationships, anomalies, and insights in the data. It is a crucial step before building models, as it provides an understanding of the data's structure and can guide subsequent analysis and modeling decisions.

Data Visualization of All Features with Their Distributions involves creating visual representations (e.g., histograms, box plots) of the distributions of individual features in a dataset. Visualizing feature distributions helps in understanding the spread, central tendency, and potential outliers in the data. It aids in identifying the nature of the variables and can guide decisions on data preprocessing and transformation.

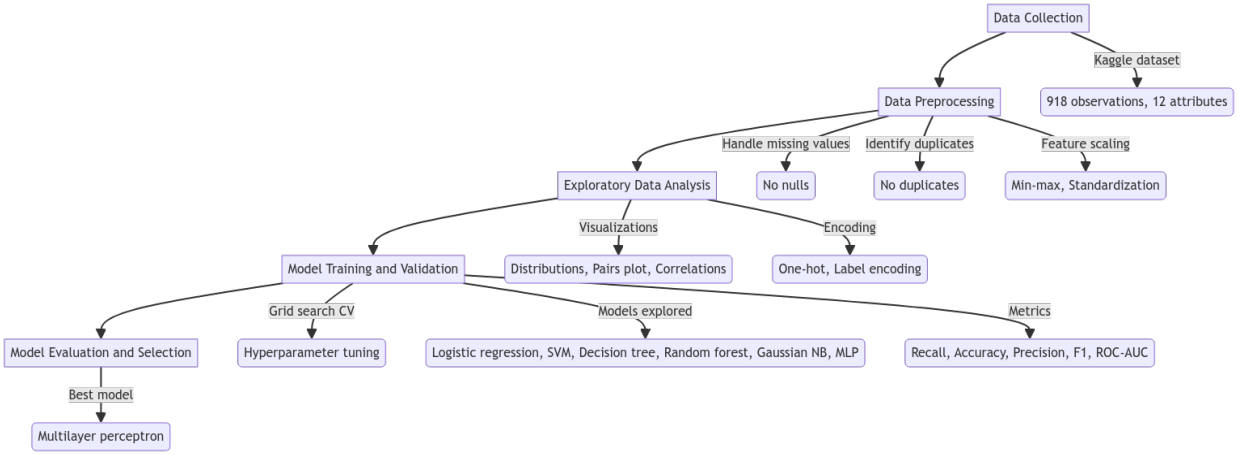
Pairwise Plots, such as scatter plots or scatterplot matrices, show the relationships between pairs of variables in a dataset. Pairwise plots help identify patterns, correlations, and potential clusters among variables. They are useful for gaining insights into multivariate relationships in the data.

Correlation Matrix, a table showing correlation coefficients between variables. Each cell in the table represents the correlation between two variables. It is important for understanding how variables are related to each other. High correlations may indicate dependencies or redundancies between features, which can impact model performance. It helps in feature selection and model interpretation.

One-Hot Encoding converts categorical variables into binary vectors (0s and 1s) to represent different categories.

Label Encoding assigns a unique numerical label to each category in a categorical variable. Machine learning algorithms often require numerical input. One-hot encoding and label encoding are techniques to represent categorical data numerically, ensuring that models can effectively use this information.

Flow chart:



4 IMPLEMENTATION

4.1 DATA PREPROCESSING:

In preparing and cleaning the dataset for heart disease prediction, we conducted thorough checks for missing values (nulls) and duplicate entries to ensure the data's quality and reliability.

Identification of Null Values:

A systematic examination of each feature was performed to identify the presence of missing values (nulls). The Pandas library in Python was employed to facilitate this examination, utilizing functions such as isnull() to detect null values.

```
In [6]: df.isnull().sum()

Out[6]: Age              0
        Sex              0
        ChestPainType    0
        RestingBP        0
        Cholesterol      0
        FastingBS        0
        RestingECG       0
        MaxHR            0
        ExerciseAngina   0
        Oldpeak          0
        ST_Slope         0
        HeartDisease     0
        dtype: int64
```

Duplicate check has been performed if there are any duplicate instances of the dataset are present.
There are 0 duplicates.

```
In [7]: df.duplicated().sum()

Out[7]: 0
```

All the numerical attributes values have been converted to float type, for easier processing.

```
In [8]: # Iterate through columns
        for col in df.columns:
            if pd.api.types.is_numeric_dtype(df[col]):
                df[col] = df[col].astype('float64')
```

```
In [9]: df.head()
```

Out[9]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAr
0	40.0	M	ATA	140.0	289.0	0.0	Normal	172.0	
1	49.0	F	NAP	160.0	180.0	0.0	Normal	156.0	
2	37.0	M	ATA	130.0	283.0	0.0	ST	98.0	
3	48.0	F	ASY	138.0	214.0	0.0	Normal	108.0	
4	54.0	M	NAP	150.0	195.0	0.0	Normal	122.0	

Feature Scaling:

Feature scaling is a crucial preprocessing step in machine learning workflows, aiming to normalize the range of distinctive features, ensuring that all variables contribute equally to model training. In our heart disease prediction study, we

employed specific scaling techniques for unique features to enhance the performance of the machine learning models.

Min-Max Scaling on 'Oldpeak':

Oldpeak, representing the ST depression, is a numeric feature with a range that can vary significantly. Min-Max scaling was applied to transform this feature, ensuring its values fall within a consistent range, typically [0, 1].

- The formula used for Min-Max scaling is given by: $X_{Scaled} = (X - \min(X))/(\max(X) - \min(X))$
- Oldpeak feature is normalized as it had displayed a right skewed data distribution.
- This transformation was performed to prevent features with larger numeric ranges from dominating the learning process.

Standard Scaling on 'Age,' 'RestingBP,' 'Cholesterol,' and 'MaxHR':

Age, Resting Blood Pressure (RestingBP), Serum Cholesterol (Cholesterol), and Maximum Heart Rate Achieved (MaxHR) are numeric features with different scales. Standard scaling (Z-score normalization) was chosen to center these variables around zero with a standard deviation of one.

- The formula used for standard scaling is given by:$X_{scaled} = X - mean(x)/ std(X)$
- Age, RestingBP, Cholesterol and MaxHR features are scaled down because these features are normally distributed.
- This scaling method ensures that features have a mean of zero and a standard deviation of one, promoting uniformity and aiding convergence during model training.

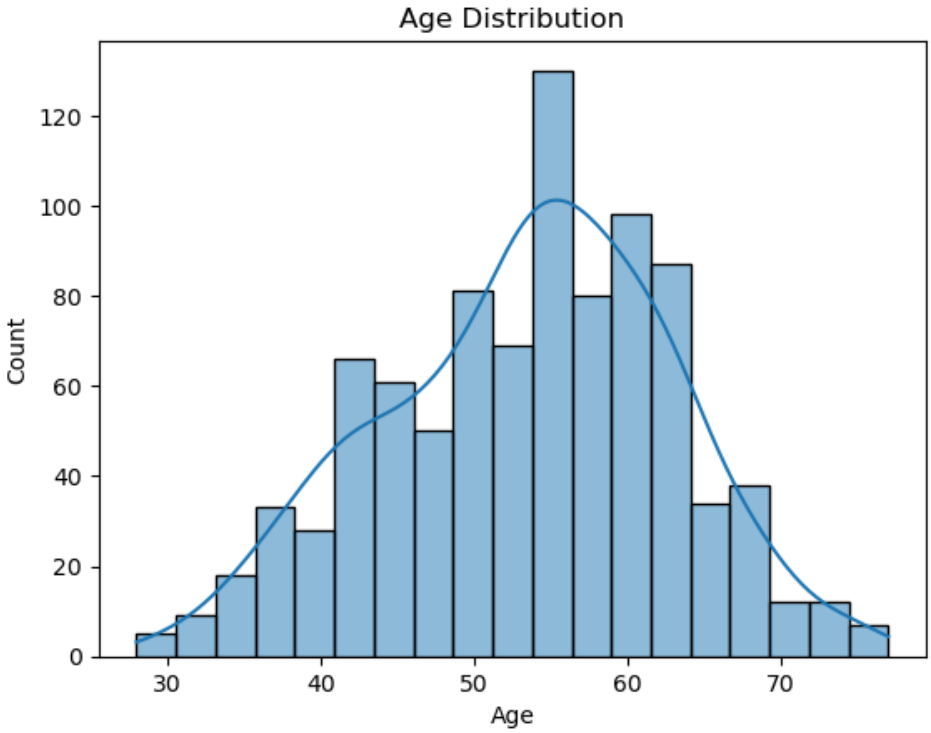
In [10]:
```
mms = MinMaxScaler()
ss = StandardScaler()

df['Oldpeak'] = mms.fit_transform(df[['Oldpeak']])
df['Age'] = ss.fit_transform(df[['Age']])
df['RestingBP'] = ss.fit_transform(df[['RestingBP']])
df['Cholesterol'] = ss.fit_transform(df[['Cholesterol']])
df['MaxHR'] = ss.fit_transform(df[['MaxHR']])

df.head()
```

Out[10]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	-1.433140	M	ATA	0.410909	0.825070	0.0	Normal	1.382928	N	0.295455	Up	0.0
1	-0.478484	F	NAP	1.491752	-0.171961	0.0	Normal	0.754157	N	0.409091	Flat	1.0
2	-1.751359	M	ATA	-0.129513	0.770188	0.0	ST	-1.525138	N	0.295455	Up	0.0
3	-0.584556	F	ASY	0.302825	0.139040	0.0	Normal	-1.132156	Y	0.465909	Flat	1.0
4	0.051881	M	NAP	0.951331	-0.034755	0.0	Normal	-0.581981	N	0.295455	Up	0.0

4.1 EXPLORATORY DATA ANALYSIS:

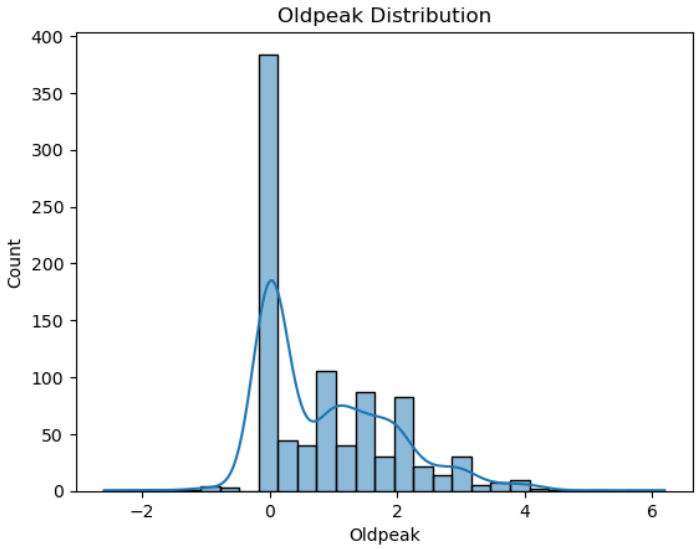
Age is normally distributed with a mean of 53.510893 and a standard distribution of 9.432617.



Full statistics of the Age features can be seen below:

```
In [13]: df['Age'].describe()

Out[13]: count    918.000000
         mean      53.510893
         std        9.432617
         min       28.000000
         25%       47.000000
         50%       54.000000
         75%       60.000000
         max       77.000000
         Name: Age, dtype: float64
```

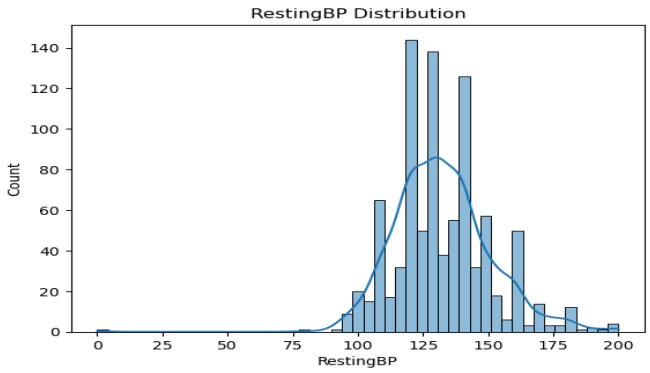
Oldpeak: Oldpeak has a right skewed distribution with a mean of 0.887364 and standard deviation of 1.066570.



Full statistics of the Oldpeak features can be seen below:

```
In [15]: df['Oldpeak'].describe()

Out[15]: count    918.000000
         mean       0.887364
         std        1.066570
         min       -2.600000
         25%        0.000000
         50%        0.600000
         75%        1.500000
         max        6.200000
         Name: Oldpeak, dtype: float64
```

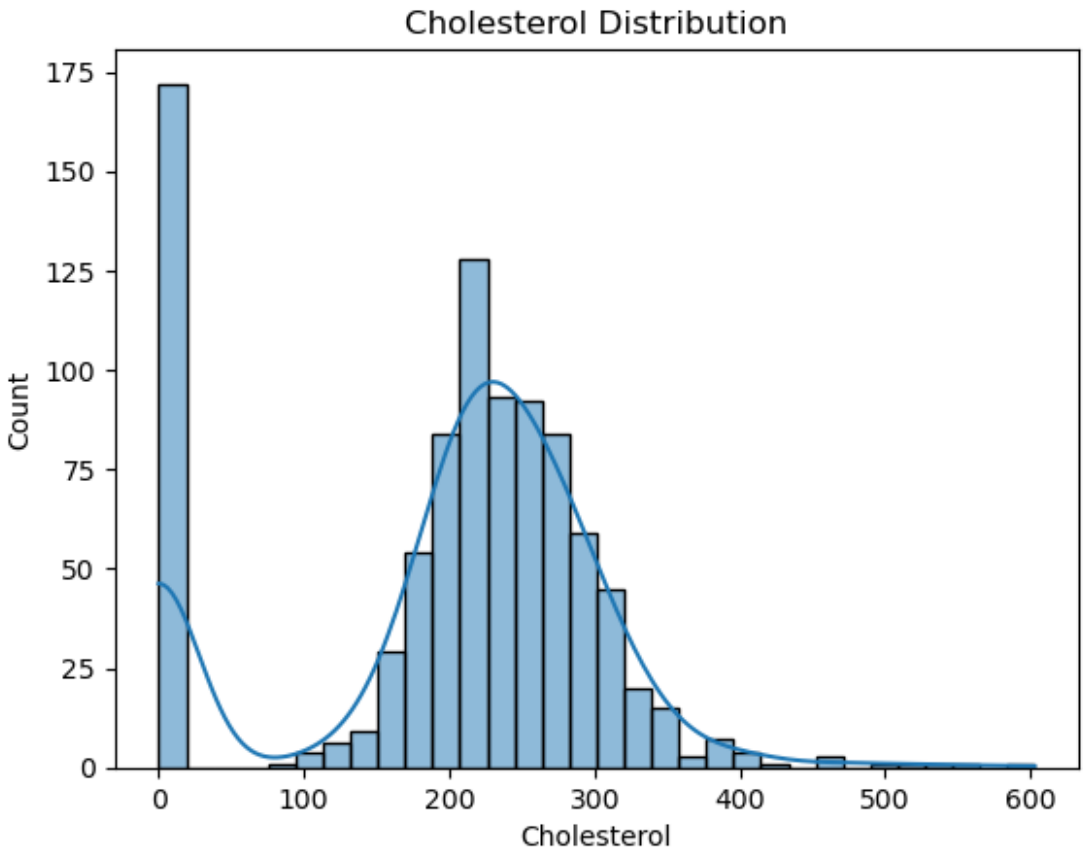
RestingBP: RestingBP is close to normal distribution with a mean of 132.396514 and standard deviation of 18.514154.



Full statistics of the RestingBP can be seen below:

```
In [17]: df['RestingBP'].describe()

Out[17]: count    918.000000
         mean     132.396514
         std       18.514154
         min        0.000000
         25%      120.000000
         50%      130.000000
         75%      140.000000
         max      200.000000
         Name: RestingBP, dtype: float64
```

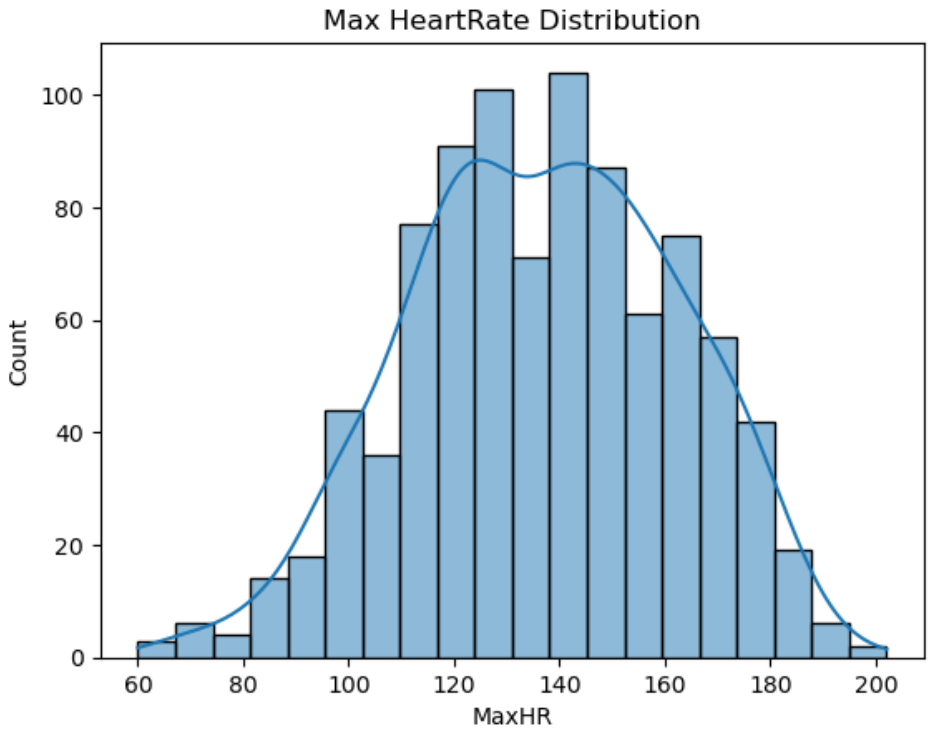
Cholesterol: Cholesterol is distributed as shown in the below figure with mean of 198.799564, standard deviation of 109.384145.



Full statistics for Cholesterol feature can be seen below:

```
In [19]: df['Cholesterol'].describe()

Out[19]: count    918.000000
         mean     198.799564
         std      109.384145
         min        0.000000
         25%      173.250000
         50%      223.000000
         75%      267.000000
         max      603.000000
         Name: Cholesterol, dtype: float64
```

MaxHR: Maximum Heart Rate is normally distributed with a mean of 136.809368 and standard deviation of 25.460334.

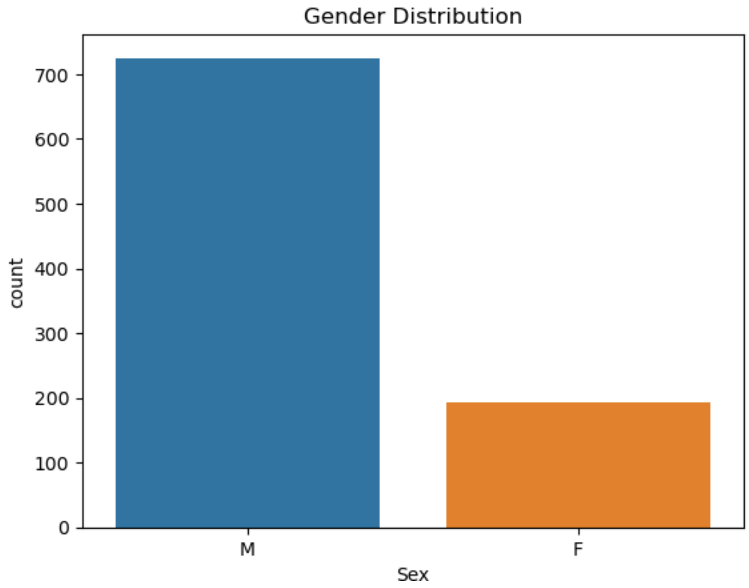


Full stats of the MaxHR feature can be seen below:

```
In [21]: df['MaxHR'].describe()

Out[21]: count    918.000000
         mean     136.809368
         std       25.460334
         min       60.000000
         25%      120.000000
         50%      138.000000
         75%      156.000000
         max      202.000000
         Name: MaxHR, dtype: float64
```

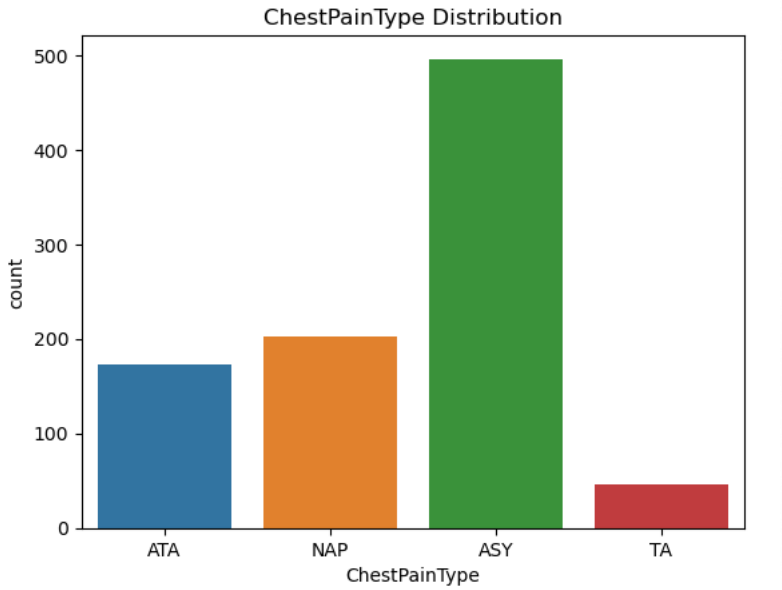
Sex: Gender of the patients is a categorical feature with two categories ‘M’ and ‘F,’ with Male being top with a frequency of 725.

```
In [23]: df['Sex'].describe()

Out[23]: count     918
         unique      2
         top         M
         freq      725
         Name: Sex, dtype: object
```



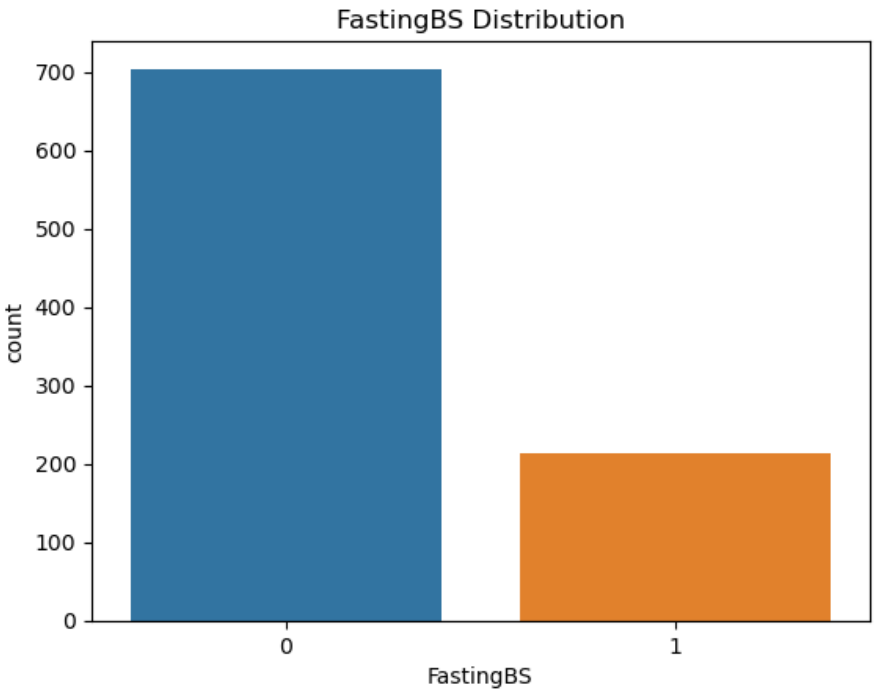
ChestPainType: Chest pain type is a categorical feature with 4 categories 'ASY','ATA','NAP' and 'TA', with 'ASY' being in the top with a frequency of 496 entries.

```
In [25]: df['ChestPainType'].describe()

Out[25]: count     918
         unique      4
         top       ASY
         freq      496
         Name: ChestPainType, dtype: object
```



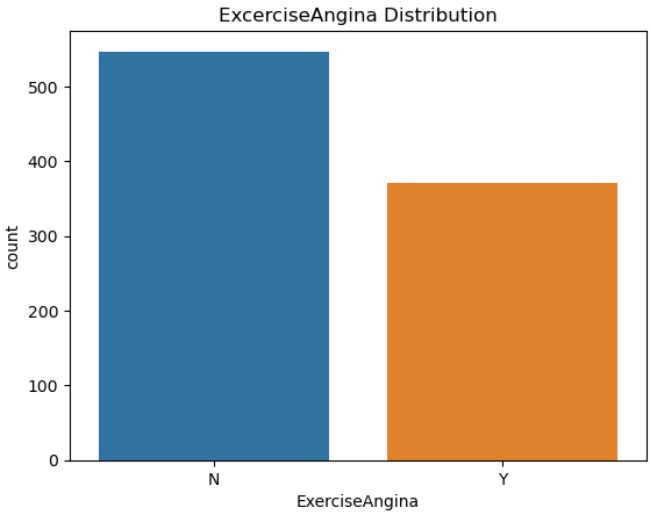
FastingBS: Fasting Blood sugar is a categorical feature with values being 0 and 1. It has the highest occurrence of 0(<120 mg/dl) with 704 occurrences in the entire data set.

```
In [29]: df['FastingBS'].astype('category').describe()

Out[29]: count     918
         unique      2
         top         0
         freq      704
         Name: FastingBS, dtype: int64
```



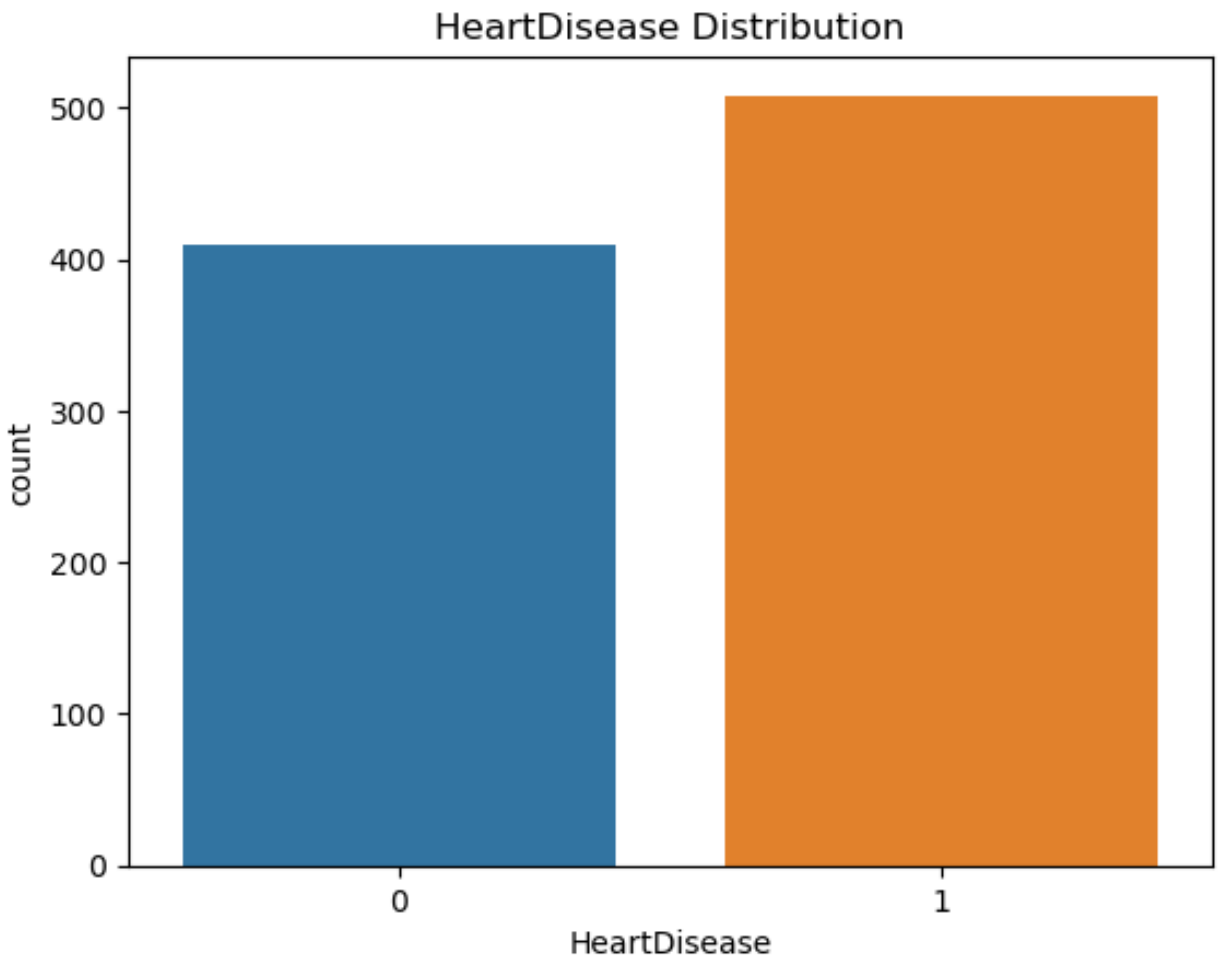
ExerciseAngina: Exercise Angina is a categorical feature with the values being Y and N. N is the top occurred value in the dataset with the frequency of 547.

```
In [30]: df['ExerciseAngina'].describe()

Out[30]: count     918
         unique      2
         top         N
         freq      547
         Name: ExerciseAngina, dtype: object
```

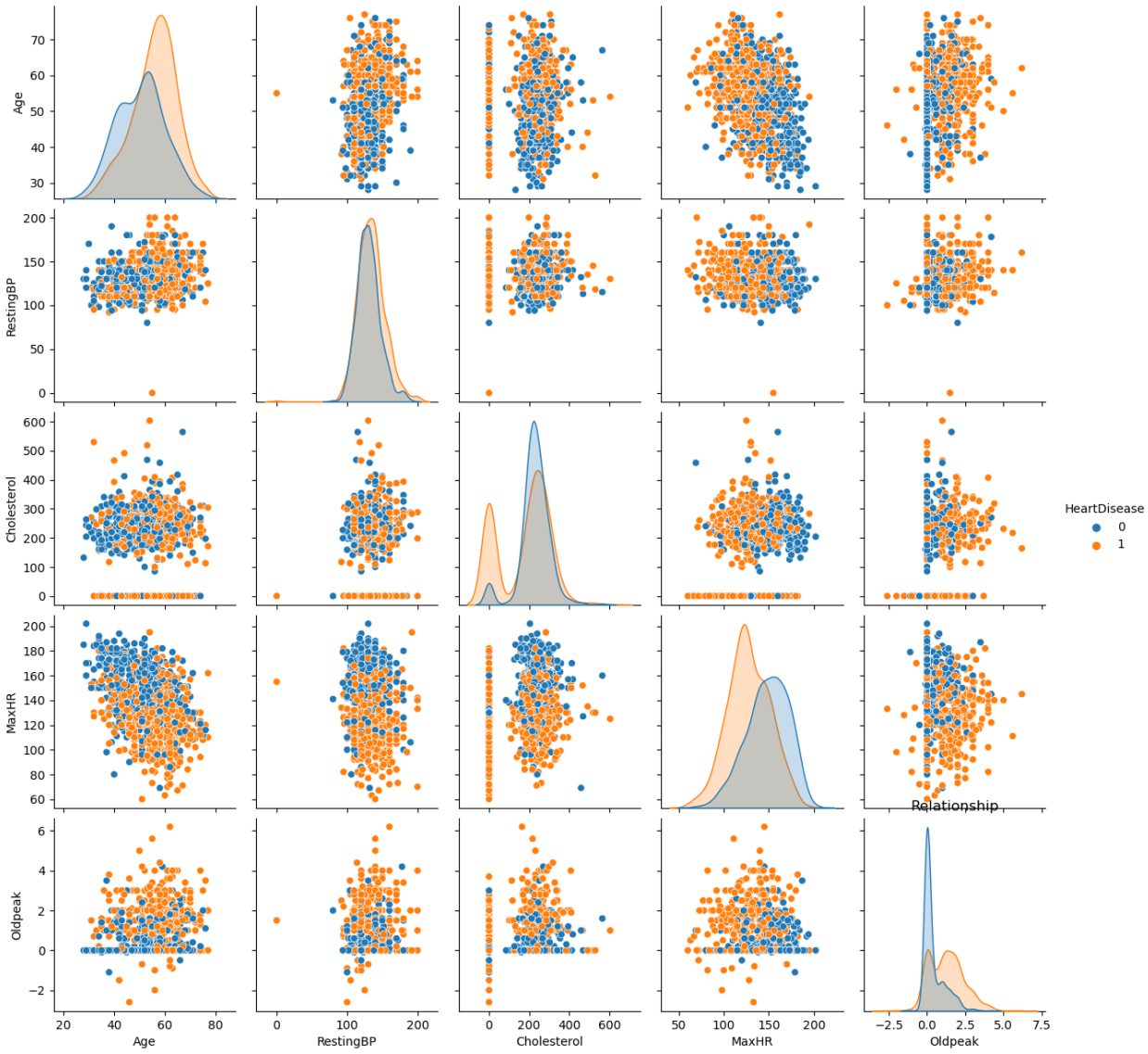


HeartDisease: Heart Disease is the target classification variable with the values being 0(Normal) and 1(heart disease). 1 occurred in 508 entries in the dataset. This indicates the dataset is balanced.

```
In [33]: df['HeartDisease'].astype('category').describe()

Out[33]: count     918
         unique      2
         top         1
         freq      508
         Name: HeartDisease, dtype: int64
```

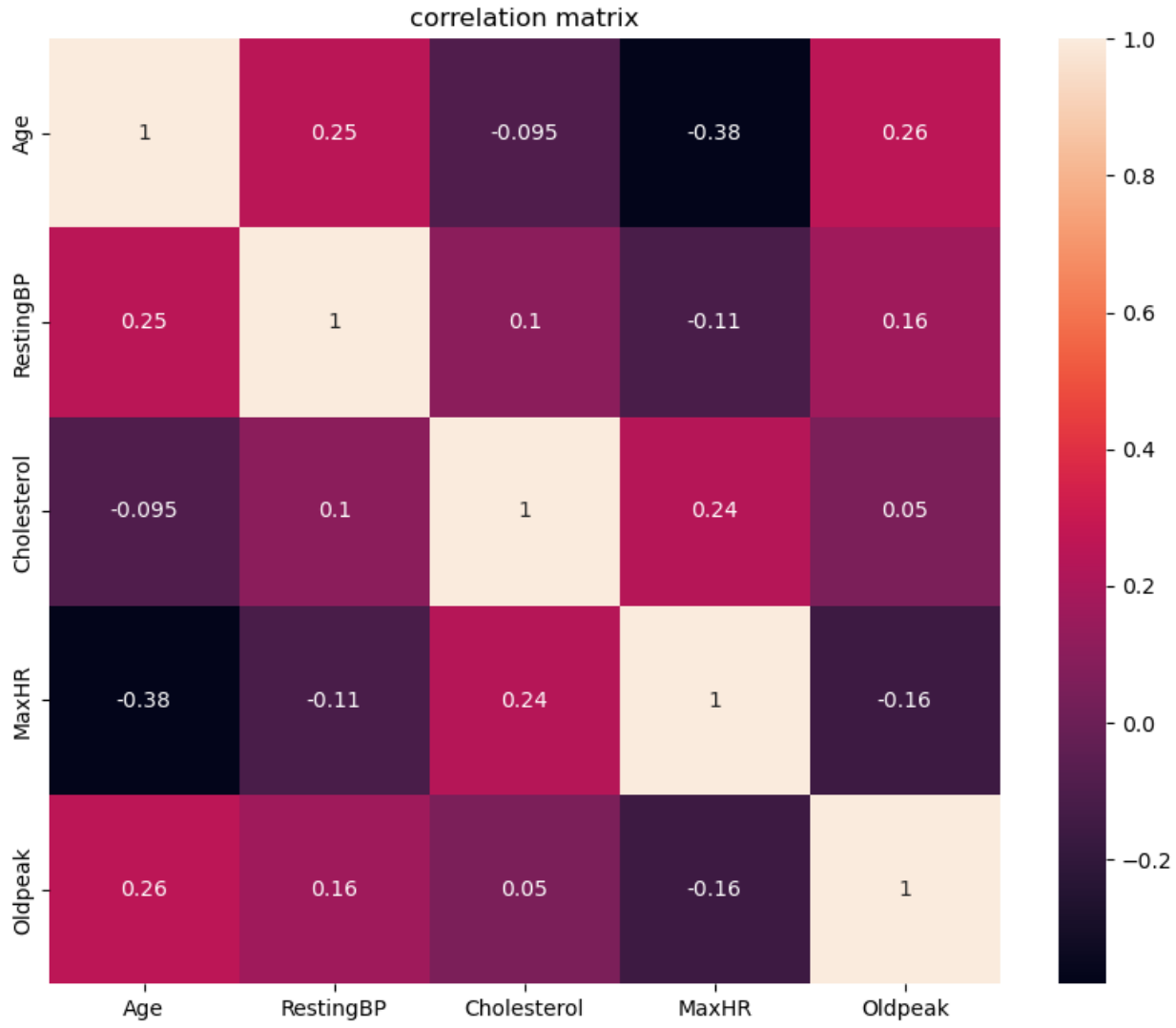


Pair plot: scatterplots help visualize the relationship or pattern between pairs of variables. Common patterns include linear relationships, clusters, outliers, or no discernible pattern.



Correlation Matrix:

The correlation matrix gives correlation coefficients, a statistical measure describing the extent to which two variables change together. It ranges from -1 to 1.



Encoding:

One hot encoding:

One-hot encoding is a technique used to convert categorical variables into a binary matrix (1s and 0s). One-hot encoding is commonly used in scenarios where the categorical variable has no inherent ordinal relationship.

One hot encoding is being performed on the below features.

- ChestPainType
- RestingECG
- ST_Slope

Every time one hot encoding is performed, it adds all categories in that feature as different ones to the data set. For example, in the case of ChestPainType, the categories are ASY, NAP, ATA, TA. All these categories will be a new feature with 1 in their data instances and 0 in others.

We removed the less occurred categories in the features from the one hot encoded data set to avoid dummy variable trap.

Label encoding: Label encoding is another technique for encoding categorical variables into numerical format. In label encoding, each unique category is assigned an integer label. This is particularly useful for ordinal categorical variables, where the order of the categories is meaningful.

Label encoding is being performed on the below features:

- Sex
- ExerciseAngina
- FastingBS

Principal Component Analysis:

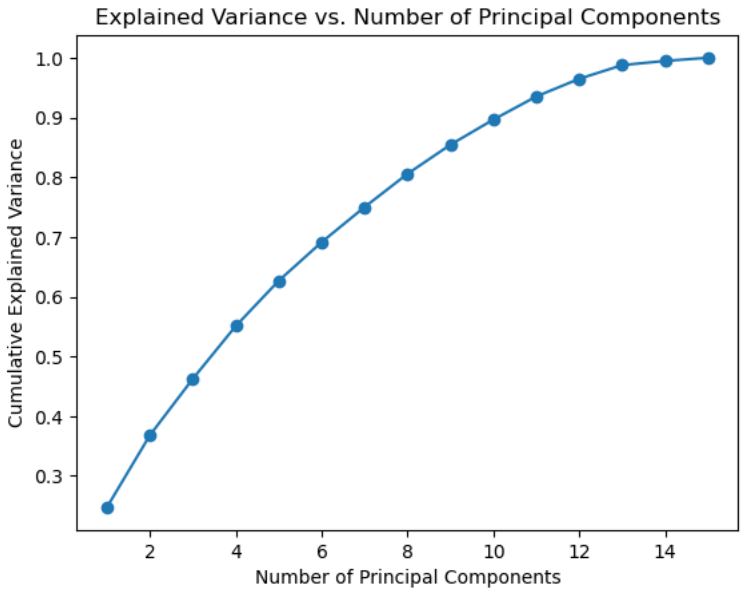
We performed PCA on the unlabeled data and plotted the relation between explained variance and number of principal components.

Cumulative Explained Variance:

Cumulative Explained Variance is the accumulation of the explained variance across multiple principal components. It is calculated by summing up the explained variances of all the principal components up to a certain component. This measure helps us understand how much of the total variance in the original data is retained when considering a specific number of principal components.
Mathematically, the Cumulative Explained Variance (CEV) at the Ith principal component is given by the formula:

$$CEV_i = \sum_{k=1}^{i} Explained\ Variance_k \,/\, \sum_{j=1}^{p} Total\ variance_j$$

Here, *i* is the number of principal components considered, and *p* is the total number of original features or dimensions in the dataset.



Since there is significant information loss for every principal component less than the total features, we did not consider doing PCA as this dataset has a smaller number of features.

The Pre-processed data:

Out[51]:

	Age	Sex	RestingBP	Cholesterol	FastingBS	MaxHR	ExerciseAngina	Oldpeak	HeartDisease	ASY	ATA	NAP	LVH	Normal	Flat	Up
0	-1.433140	1	0.410909	0.825070	0	1.382928	0	0.295455	0	0	1	0	0	1	0	1
1	-0.478484	0	1.491752	-0.171961	0	0.754157	0	0.409091	1	0	0	1	0	1	1	0
2	-1.751359	1	-0.129513	0.770188	0	-1.525138	0	0.295455	0	0	1	0	0	0	0	1
3	-0.584556	0	0.302825	0.139040	0	-1.132156	1	0.465909	1	1	0	0	0	1	1	0
4	0.051881	1	0.951331	-0.034755	0	-0.581981	0	0.295455	0	0	0	1	0	1	0	1

4.3 MODEL TRAINING AND VALIDATION

In the pursuit of developing an effective heart disease prediction model, a comprehensive model training and evaluation pipeline was executed. This involved the exploration of multiple machine learning algorithms, each fine-tuned with Grid Search Cross-Validation to identify optimal hyperparameters. The models considered for this study encompassed Logistic Regression, Support Vector Classifier (SVC), Decision Tree Classifier, Random Forest Classifier, Naive Bayes Classifier (GaussianNB), and a Multilayer Perceptron (MLP).

Model selection:

Logistic Regression:

Logistic regression is a statistical method used for binary classification, which means predicting the probability of an observation belonging to one of two classes. Despite its name, logistic regression is a classification algorithm rather than a regression algorithm.

Parameters explored:

- Regularization strength (C) - The parameter C in logistic regression represents the inverse of the regularization strength. A smaller C value increases regularization, and a larger C value reduces it. The choice of C influences the trade-off between fitting the training data well and keeping the model simple.
- Solver - The solver in logistic regression refers to the optimization algorithm used to find the coefficients that minimize the logistic loss function. Common choices include 'liblinear', 'newton-cg', 'lbfgs', 'sag', and 'saga'. The selection of the solver depends on factors such as the size of the dataset and desired optimization properties.
- Penalty - The penalty parameter in logistic regression specifies the type of regularization applied. The two common choices are 'l1' and 'l2', corresponding to L1 and L2 regularization, respectively. 'l1' regularization promotes sparsity in the model, while 'l2' regularization penalizes large coefficients.

Support vector classifier:

A Support Vector Classifier (SVC), also known as Support Vector Machines (SVM) when used for classification, is a supervised machine learning algorithm that aims to find a hyperplane in a high-dimensional space that best separates data points

from different classes. The goal is to create a decision boundary that maximally separates instances of one class from another.

Parameters explored:

- Kernel type - The kernel in an SVM is a function that takes input data points and transforms them into a higher-dimensional space, making it easier to find a hyperplane that separates the data. Common kernel types include:

1)Linear Kernel (linear): No transformation, a linear decision boundary in the original feature space.
2)Radial Basis Function (RBF) Kernel (rbf): Often the default, suitable for non-linear boundaries.
3)Polynomial Kernel (poly): Allows modeling of non-linear relationships through polynomial transformations.
4)Sigmoid Kernel (sigmoid): Like the sigmoid function in logistic regression.

- Regularization parameter(C) - The regularization parameter, often denoted as C, controls the trade-off between having a smooth decision boundary and classifying the training points correctly. A smaller C encourages a smoother boundary, and a larger C allows the model to focus more on correctly classifying the training points.
- Gamma - The gamma parameter defines how far the influence of a single training example reaches. It affects the shape of the decision boundary. A small gamma creates a wider decision boundary, while a large gamma leads to a more complex, narrower decision boundary.

Decision Tree Classifier:

A Decision Tree Classifier is a supervised machine learning algorithm used for both classification and regression tasks. It works by recursively partitioning the data into subsets based on the values of input features. The decision tree structure is akin to a flowchart, where each internal node represents a decision based on a feature, each branch represents an outcome of that decision, and each leaf node represents the final predicted class or numerical value.

Parameters explored:

- Criterion - The criterion is the function used to measure the quality of a split at each node in the decision tree. Common criteria include:
    - **Gini impurity (gini):** Measures the frequency at which a randomly chosen element would be incorrectly classified.

- **Entropy (entropy):** Measures the level of impurity or disorder in a set. It is based on the concept of information gain.
- **Classification Error (classification error):** Measures the fraction of training observations that are misclassified at a given node.

The choice of criterion depends on the problem at hand. Gini impurity and entropy are more commonly used in practice.

- Maximum Depth - Maximum depth is the maximum depth of the decision tree. It restricts the number of nodes or levels in the tree. A shallow tree (low maximum depth) is less complex and may be less prone to overfitting, while a deep tree (high maximum depth) may capture more complex patterns in the training data but is at a higher risk of overfitting and capturing noise.
- Minimum sample split - Minimum sample split is the minimum number of samples required to split an internal node. If the number of samples at a node is less than the specified minimum, the node will not be split, and it will become a leaf node. This parameter helps control the granularity of the tree and can prevent the model from creating nodes for small subsets of the data, which might be noisy.

Random Forest Classifier:

A Random Forest Classifier is an ensemble learning method based on the construction of multiple decision trees during training and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random Forests are a popular and powerful machine learning algorithm known for their high accuracy, robustness, and ability to handle complex data sets.

Parameters Explored:

Number of trees in the forest(n_estimators) - This parameter determines how many individual decision trees are trained and combined to make predictions. The larger the number of trees, the more robust the Random Forest may become, but it also comes with increased computational cost.

The formula for the prediction in a Random Forest can be summarized as follows:

Regression: For regression problems, the prediction is often the average (or sometimes the median) of the predictions from all the individual trees in the forest: $y\hat{} = N1 \sum (i=1 \text{to } N)\ yi$ where $y\hat{}$ is the predicted value, N is the number of trees, and yi is the prediction of the i-th tree.

Classification: For classification problems (where each tree votes for a class), the predicted class is determined by a majority vote:
$y^ = \text{argmax} \sum_{i=1}^{N} I(y_i = y)$ where $y^$ is the predicted class, N is the number of trees, y_i is the predicted class of the i-th tree, and $I(\cdot)$ is the indicator function.
Regarding the n_estimators parameter, increasing its value improves the performance of the Random Forest up to a certain point. After a certain number of trees, the improvement may become marginal, and there might be diminishing returns or even overfitting. The optimal value for n_estimators depends on the specific dataset and problem.

Naïve Bayes classifier (Gaussian NB):

Naïve Bayes, including the Gaussian Naïve Bayes (GNB) variant, is a probabilistic machine learning algorithm based on Bayes' theorem. It is particularly useful for classification tasks and is known for its simplicity, efficiency, and effectiveness, especially when dealing with high-dimensional data.

No hyperparameter tuning, as Gaussian Naïve Bayes is not sensitive to hyperparameters.

Multilayer Perceptron (MLP):

Multilayer Perceptron (MLP) is a type of artificial neural network that falls under the category of deep learning models. It is a feedforward neural network, meaning that information travels in one direction—from input to output—through a series of layers.

Parameters explored:

- Hidden layer sizes - The hidden layer sizes specify the number of neurons (nodes) in each hidden layer of the neural network. You can have multiple hidden layers, and the hidden layer sizes parameter defines the number of neurons in each layer. The architecture of the hidden layers influences the capacity of the neural network to learn complex patterns.
- Activation - The activation function is used to introduce non-linearity to the network. Common activation functions include:
    - **Rectified Linear Unit (ReLU):** $f(x)=\max(0, x)$
    - **Sigmoid:** $f(x)=1/1+e^x$
    - **Hyperbolic Tangent (tanh):** $f(x)=\tanh(x)$

- The choice of activation function depends on the problem and the data's characteristics.

- Regularization value - Regularization is used to prevent overfitting by penalizing large weights in the network. The regularization value controls the strength of regularization. Common types include L1 regularization and L2 regularization. The regularization term is added to the loss function during training.
- Learning Rate: The learning rate determines the step size taken during optimization when adjusting the network's weights. Too high a learning rate may cause the model to converge quickly but might overshoot the minimum, while too low a learning rate may lead to slow convergence. Learning rates are often set to small values, such as 0.1, 0.01, or smaller.
- Max iterations -
    - The maximum number of iterations (epochs) defines how many times the entire training dataset is passed forward and backward through the neural network. Training stops when this number of iterations is reached or when convergence is achieved. It helps prevent the model from training for too long, especially when there is a risk of overfitting.

Grid search CV

```
In [ ]: model_params = {
    'decision_tree': {
        'model': DecisionTreeClassifier(criterion = 'entropy',random_state=42),
        'params':{
            'max_depth': [3,5,7],
            'min_samples_split': [2,5,10]
        }
    },
    'naive_bayes': {
        'model': GaussianNB(),
        'params': {
            'var_smoothing': [1e-9, 1e-8, 1e-7, 1e-6, 1e-5]
        }
    },

    'svm': {
        'model': SVC(gamma = 'auto'),
        'params': {
            'C': [1,10,20],
            'kernel': ['rbf','linear']
        }
    },
    'random_forest':{
        'model': RandomForestClassifier(),
        'params': {
            'n_estimators': [1,5,10]
        }

    },
    'logistic_reg': {
        'model': LogisticRegression(solver = 'liblinear',multi_class='auto'),
        'params': {
            'C': [1,5,10]
        }

    },
    'mlp': {
        'model': MLPClassifier(),
        'params': {
            'hidden_layer_sizes': [(10,), (50,), (100,)],
            'activation': ['relu', 'tanh', 'logistic'],
            'alpha': [0.0001, 0.001, 0.01],
            'learning_rate': ['constant', 'invscaling', 'adaptive'],
            'learning_rate_init': [0.001, 0.01, 0.1],
            'max_iter': [100, 200, 300]
        }
    }
}
```

Grid Search Cross-Validation:

Procedure:

- A Grid Search Cross-Validation approach was adopted to systematically explore hyperparameter combinations for each model.
- A 3-fold cross-validation setup (K=3) was used, where the dataset was divided into three subsets, and the model was trained and validated three times, rotating through each subset as the validation set.

Performance Metric:

- The recall score was selected as the primary performance metric, given the importance of identifying true positives (instances of heart disease) while minimizing false negatives.

Model training and recall score collection:

Training Procedure:

- Each model was trained on the training set using the hyperparameters identified through Grid Search CV.
- The training process involved iteratively adjusting model parameters to minimize the chosen evaluation metric, focusing on maximizing recall.

```
In [ ]: kf = KFold(n_splits=3, shuffle=True, random_state=42)
```

```
In [ ]: scorer = make_scorer(recall_score, average='weighted')  # You can adjust the 'average' parameter if needed


        scores = []

        for model_name, mp in model_params.items():
            clf = GridSearchCV(mp['model'], mp['params'], cv=kf,scoring=scorer, return_train_score=False)
            clf.fit(X,y)
            best_recall = clf.best_score_
            scores.append({
                'model': model_name,
                'best_score': clf.best_score_,
                'best_params': clf.best_params_,
                'best_recall': best_recall
            })
```

```
In [ ]: df_scores = pd.DataFrame(scores, columns=['model','best_score','best_params'])
```

Recall score collection:

- The recall scores for each model were recorded after training, providing insights into their ability to correctly identify instances of heart disease.

```
In [59]: df_scores = pd.DataFrame(scores, columns=['model','best_score','best_params'])
```

```
In [60]: from IPython.display import display
         display(df_scores)
```

	model	best_score	best_params
0	decision_tree	0.831155	{'max_depth': 3, 'min_samples_split': 2}
1	naive_bayes	0.863834	{'var_smoothing': 1e-09}
2	svm	0.867102	{'C': 1, 'kernel': 'rbf'}
3	random_forest	0.836601	{'n_estimators': 10}
4	logistic_reg	0.862745	{'C': 1}
5	mlp	0.872549	{'activation': 'relu', 'alpha': 0.001, 'hidden...

```
In [61]: df_scores['best_params'][5]
Out[61]: {'activation': 'relu',
          'alpha': 0.001,
          'hidden_layer_sizes': (10,),
          'learning_rate': 'constant',
          'learning_rate_init': 0.01,
          'max_iter': 200}
```

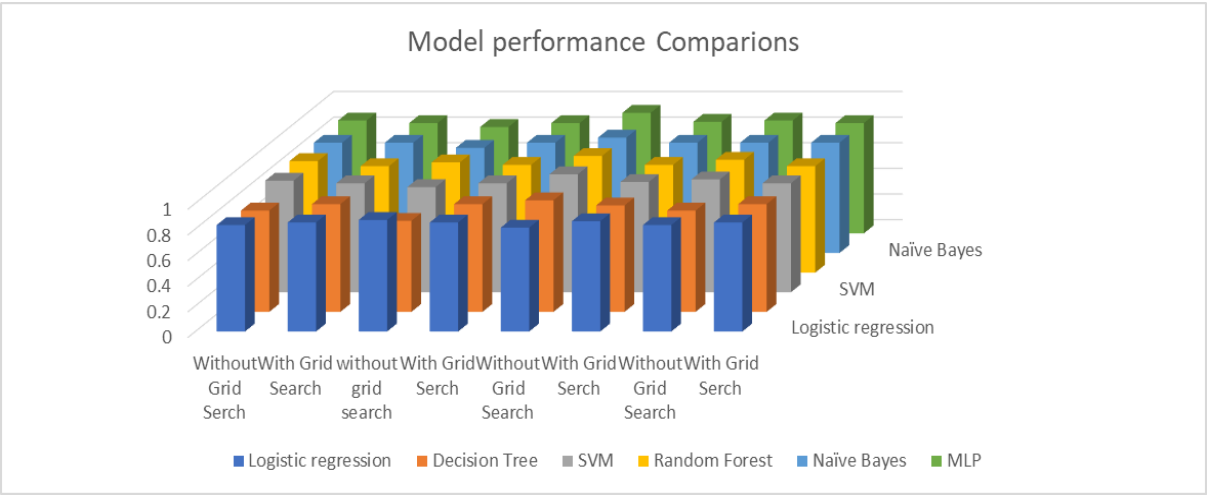
5 EXPERIMENTAL RESULTS AND DISCUSSION

The model exhibiting the highest recall score was selected as the best-performing model for heart disease prediction. Recall was prioritized due to the critical nature

of identifying true positives in a healthcare context. In the above experiment with different models with different hyper parameters, multi-layer perceptron has the best recall score of 0.8725, the logistic regression has a recall of 0.86, Decision tree and Random Forest has 0.83, 0.84, SVM has recall of 0.86 and naive bayes 0.86. As multi-layer perceptron was deemed most suitable for its ability to minimize false negatives, ensuring a more accurate identification of individuals at risk of heart disease.

The table below shows the comparison between the performance metrics of different models. The ROC-AUC curves are for the models after grid search. The differences in metrics, before and after grid search can be seen,

Model	Accuracy	Accuracy	Precision	Precision	Recall	Recall	F1 Score	F1 Score
	Without Grid Serch	With Grid Search	without grid search	With Grid Serch	Without Grid Search	With Grid Serch	Without Grid Search	With Grid Serch
Logistic regression	0.83	0.85	0.87	0.85	0.81	0.86	0.83	0.85
Decision Tree	0.79	0.84	0.71	0.84	0.87	0.83	0.79	0.84
SVM	0.87	0.85	0.82	0.85	0.92	0.86	0.88	0.85
Random Forest	0.87	0.83	0.86	0.84	0.91	0.84	0.88	0.83
Naïve Bayes	0.86	0.86	0.82	0.86	0.9	0.86	0.86	0.86
MLP	0.88	0.86	0.83	0.86	0.94	0.87	0.88	0.86



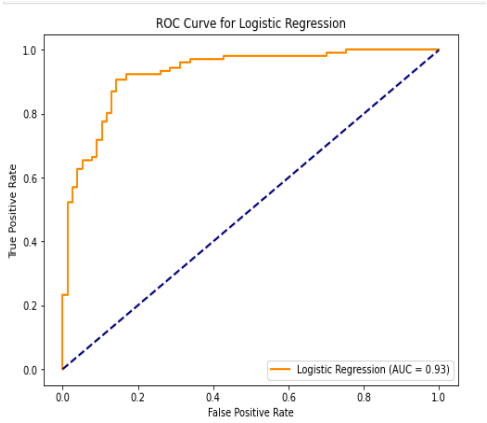
Model Evaluation:

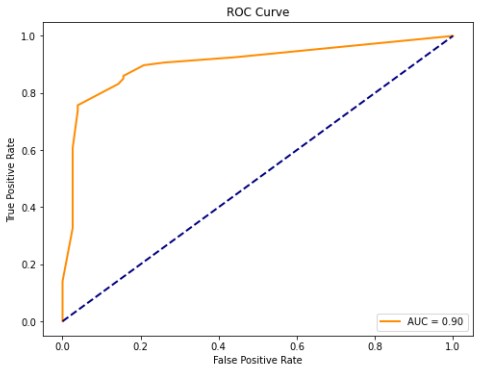
In addition to recall, other metrics such as accuracy, precision, F1 score, and ROC-AUC were computed to provide a comprehensive evaluation of each model's performance.

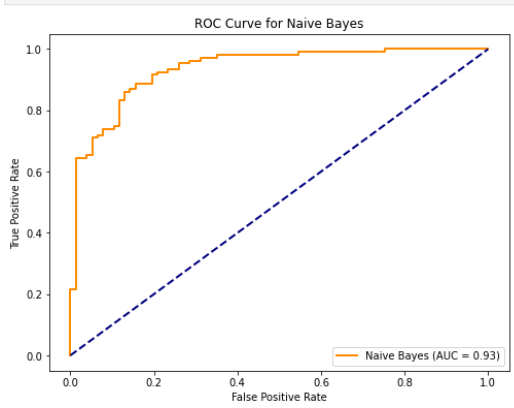
Cross-validation ensured the robustness of the chosen model by assessing its performance across different subsets of the dataset.

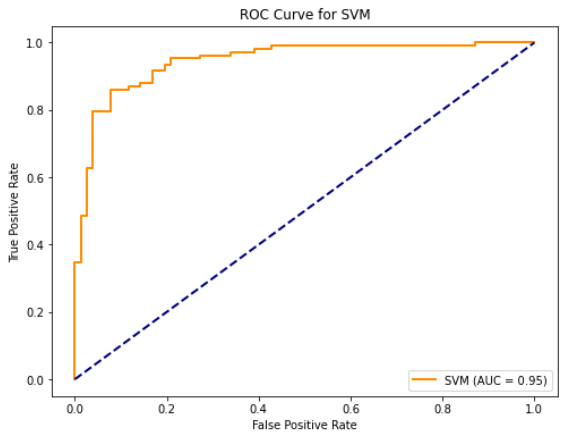
The model's ability to generalize to unseen data was also assessed using the test set.

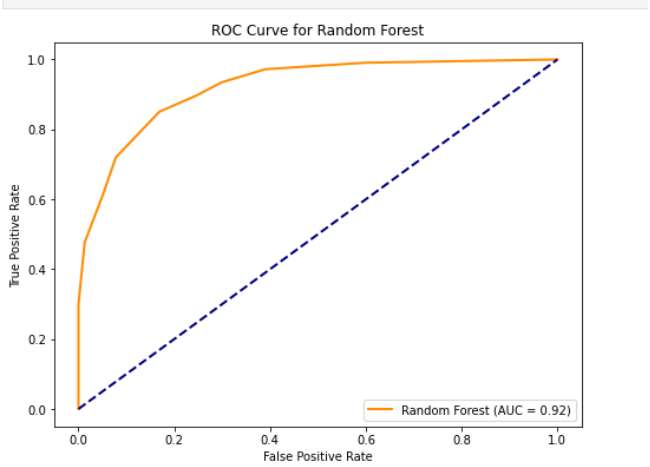
The ROC Curve is for the data with grid search,

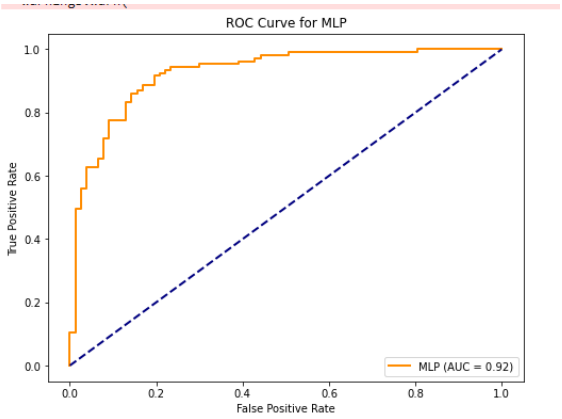


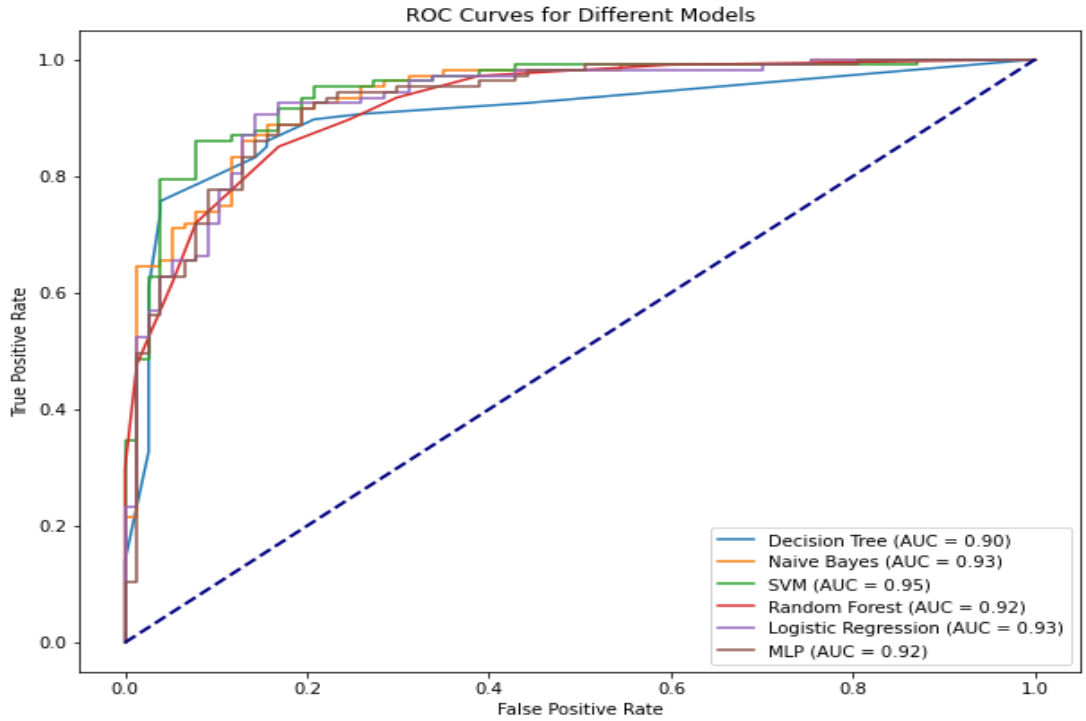












6.Conclusion: Considering the ROC-AUC Curves and the recall score, for this data set, the multi-layer perceptron has performed better than other models.

Future Direction:

In the future, we can combine two machine learning algorithms, build a hybrid model to achieve better performance leveraging OR gate or XOR gate techniques. The building of chat-bots will be helpful in the medical sector, where we can use the BERT/T5 hugging face transformers aligning with ML algorithms to make user

communication easier and achieving better recall. Implementing block chain technology to the chatbot interface to make sure privacy is secured.

References:

1 A. H. Chen, S. Y. Huang, P. S. Hong, C. H. Cheng and E. J. Lin, "HDPS: Heart disease prediction system," 2011 Computing in Cardiology, Hangzhou, China, 2011, pp. 557-560.

2 Soni, Jyoti, Ujma Ansari, Dipesh Sharma, and Sunita Soni. "Predictive data mining for medical diagnosis: An overview of heart disease prediction." International Journal of Computer Applications 17, no. 8 (2011): 43-48.

3 S. Palaniappan and R. Awang, "Intelligent heart disease prediction system using data mining techniques," 2008 IEEE/ACS International Conference on Computer Systems and Applications, Doha, Qatar, 2008, pp. 108-115, doi: 10.1109/AICCSA.2008.4493524.

4 A. Singh and R. Kumar, "Heart Disease Prediction Using Machine Learning Algorithms," 2020 International Conference on Electrical and Electronics Engineering (ICE3), Gorakhpur, India, 2020, pp. 452-457, doi: 10.1109/ICE348803.2020.9122958.

5 S. Mohan, C. Thirumalai and G. Srivastava, "Effective Heart Disease Prediction Using Hybrid Machine Learning Techniques," in IEEE Access, vol. 7, pp. 81542-81554, 2019, doi: 10.1109/ACCESS.2019.2923707.