MODEL FOR PREDICTING HEART DISEASE

**Author: Nguyen Hung Thanh**

Ho Chi Minh City, May 01, 2024

**MODEL FOR PREDICTING HEART DISEASE**

**Abstract**

This is my self-learning project. The dataset is collected from Kaggle, containing 11 common features curated by combining 5 popular heart disease datasets already available independently but not combined before.These datasets were collected and combined at one place to help advance research on CAD-related machine learning and data mining algorithms, and hopefully to ultimately advance clinical diagnosis and early treatment.

From this dataset, I plan to visualize and conduct analytical searches to determine the correlation of each feature with heart disease. My goal is to identify the most important features for predicting whether someone will have heart disease or not. Subsequently, I aim to build a model capable of predicting the likelihood of heart disease based on CAD status.

Index

|]4[1. Introduction 2](#_Toc165479378)

[1.1. Purpose of document implementation 2](#_Toc165479379)

[1.2. The goal of the subject 2](#_Toc165479380)

[1.3. Methodology used 3](#_Toc165479381)

[2. Data Collection and Preprocessing 4](#_Toc165479382)

[2.1. Data Collection 4](#_Toc165479383)

[2.2. Preprocessing Data 5](#_Toc165479384)

[2.3. Analysis and Feature engineering 7](#_Toc165479385)

[3. Methodology 25](#_Toc165479386)

[3.1. Logistic Regression 25](#_Toc165479387)

[3.2. Support Vector Machines (SVM) 26](#_Toc165479388)

[3.3. Random Forest Classification 28](#_Toc165479389)

[3.4. K-Nearest Neighbors 29](#_Toc165479390)

[4. Model Development 30](#_Toc165479391)

[5. Model Development 31](#_Toc165479392)

[6. Results 31](#_Toc165479393)

[7. Conclusion 32](#_Toc165479394)

# Introduction

## Purpose of document implementation

This document is created to describe all my analysis and knowledge that I've researched through the dataset about heart disease, which is curated by combining five popular heart disease datasets already available independently but not combined before. From that, we can see the importance of each feature in predicting whether someone has heart disease or not. Furthermore, we can predict who is at risk of heart disease based on their CAD health indicators and ultimately advance clinical diagnosis and early treatment.

## The goal of the subject

From this dataset, I plan to visualize and conduct analytical searches to determine the correlation of each feature with heart disease. My goal is to identify the most important features for predicting whether someone will have heart disease or not. Subsequently, I aim to build a model capable of predicting the likelihood of heart disease based on CAD status.

## Methodology used

A quick check through the dataset reveals that there are no null values. There appear to be a few duplicated instances, so I removed them and engineered some features that I thought would help in my analysis and model-building. I used a preprocessing tool such as StandardScaler to rescale the data.

I also utilized the Matplotlib and Seaborn libraries to visualize and analyze the dataset. By examining the plots generated from these libraries, I gained further insights into the data. Lastly, I used models from Scikit-learn to train, select the best model, and fine-tune it using GridSearchCV for improved predictive performance.

Because I dropped some unrealistic instances during the data cleaning stage, the dataset seems quite balanced, so I don't have to do anything further to balance it.

# Data Collection and Preprocessing

## Data Collection

The dataset was collected from Kaggle and contains 11 common features curated by Mr. MEXWELL by combining five popular heart disease datasets that were previously available independently.

The five datasets used for its curation are:

* Cleveland
* Hungarian
* Switzerland
* Long Beach VA
* Statlog (Heart) Data Set.

In total, the dataset consists of 1190 instances with 11 features. The purpose of curating these datasets into one unified dataset is to facilitate research on CAD-related machine learning and data mining algorithms. It is hoped that this combined dataset will ultimately advance clinical diagnosis and early treatment of heart disease.

*Here is the link to the data source:* [*https://www.kaggle.com/datasets/mexwell/heart-disease-dataset/data*](https://www.kaggle.com/datasets/mexwell/heart-disease-dataset/data)

## Preprocessing Data

The features description of the dataset is noted clearly in [documentation.pdf](../Heart_Disease/documentation.pdf). However, for convenience, I've made a few edit:

'chest\_pain':

* 'asymptomatic': 0
* 'non-anginal\_pain': 1
* 'atypical\_angina': 2
* 'typical\_angina': 3

'ST\_slope':

* downsloping: -1
* flat: 0
* upsloping: 1

'sex':

* 'male': 0
* 'female': 1

The dataset comprises 1190 instances, each associated with one of two labels representing whether they have heart disease or not. Each instance contains 11 features. As mentioned before, I've dropped some unrealistic instances and engineered 2 more features to enhance my work. Additionally, I've dropped duplicate instances, resulting in a dataset with only 746 usable instances and a total of 13 features, including parameters such as "age", "max\_heart\_rate", "cholesterol", "oldpeak", and others.

To start, I use a countplot to visualize the distribution of the label column in the dataset.

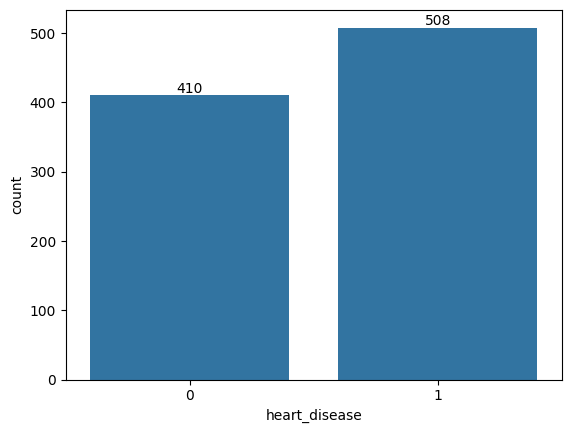


Figure 2‑1: Labels Countplot before cleaning

The countplot of labels clearly illustrates all the label names and the number of instances for each label. From this visualization, we can observe the imbalance in the dataset. Let's use a histplot and boxplot to examine the distributions of each feature to see if there are any issues with the dataset.

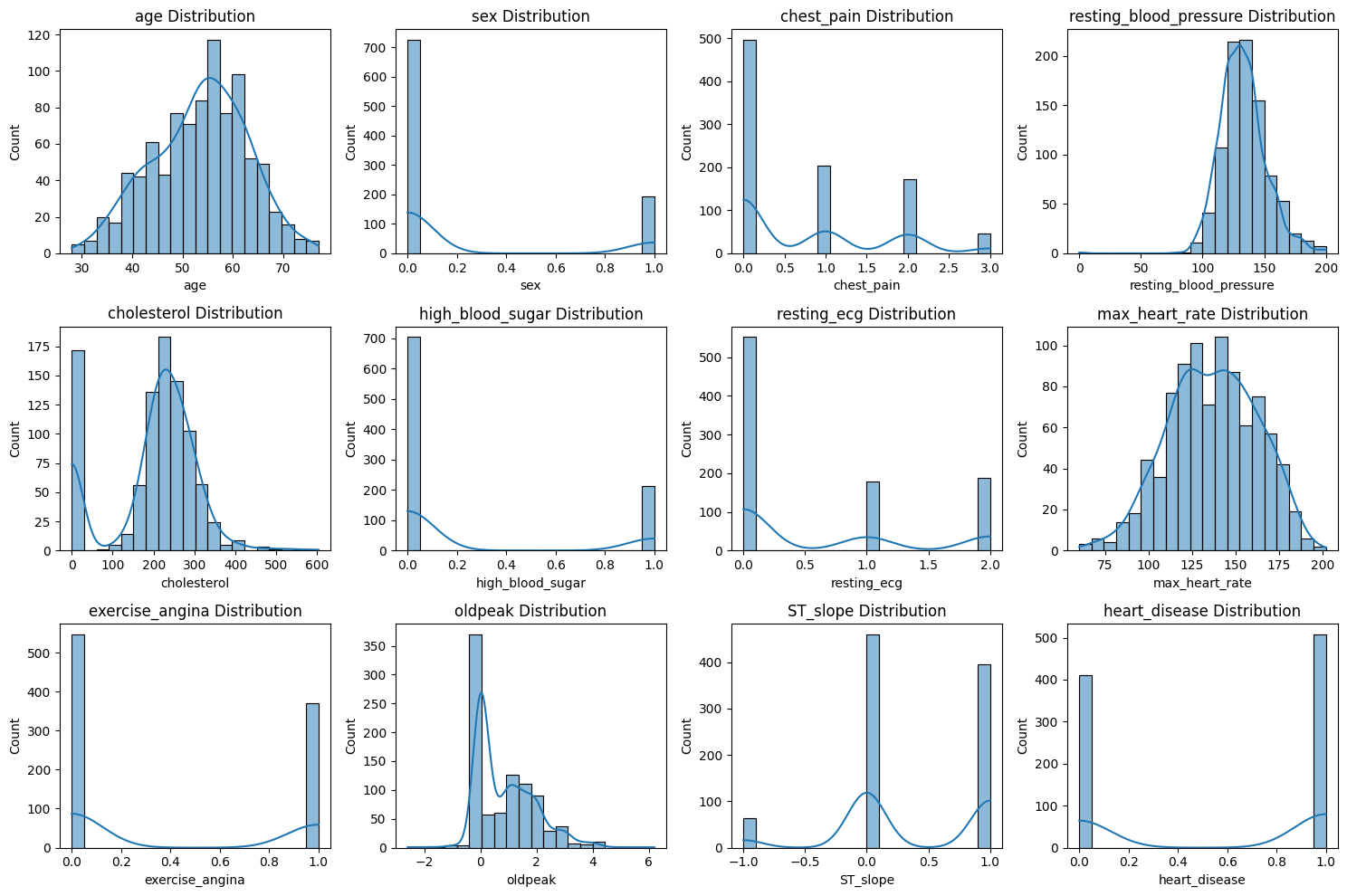


Figure 2‑2: Histplot Distributions before cleaning

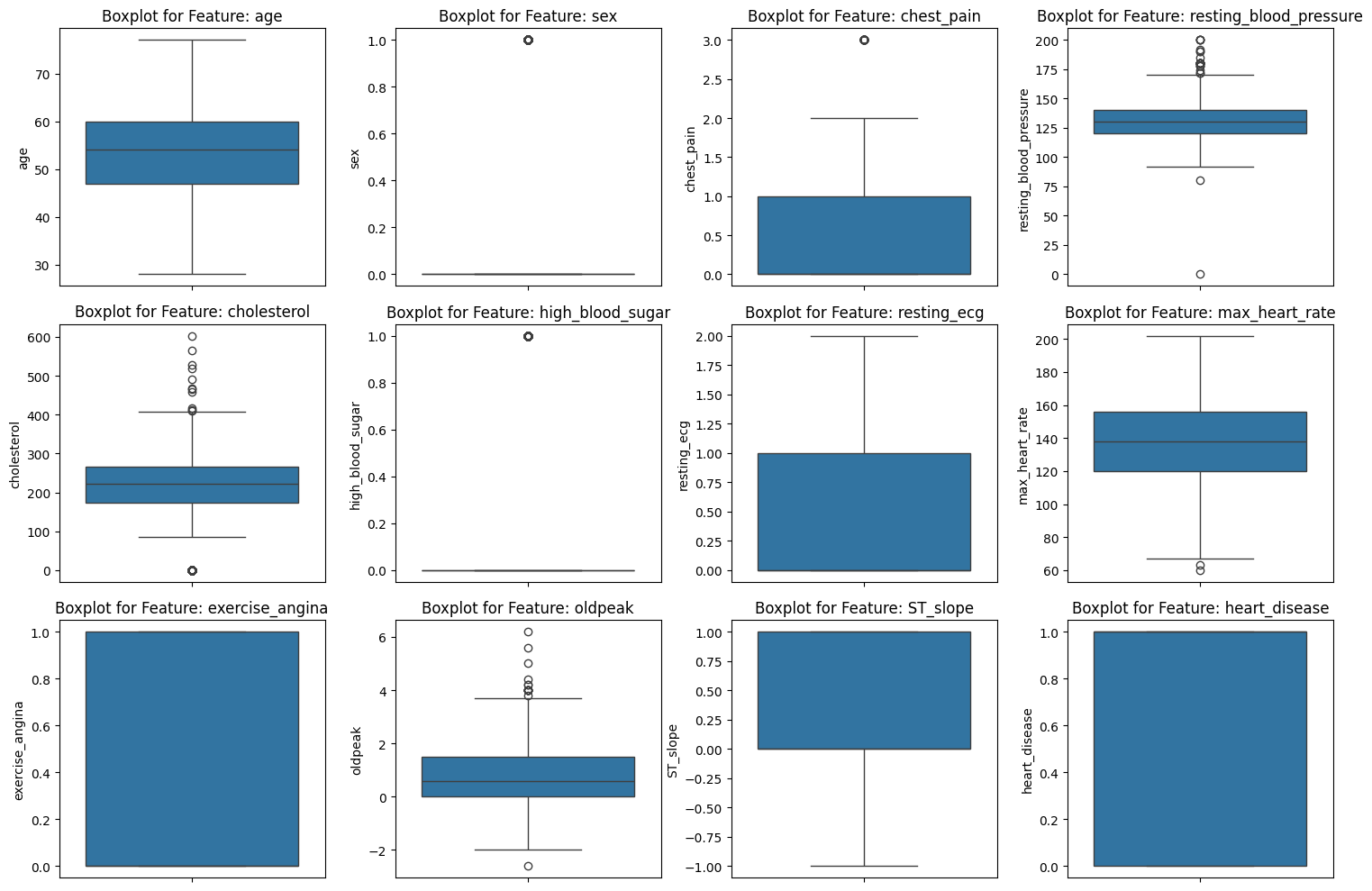


Figure 2‑3: Boxplot Distributions before cleaning

In general, the distribution of features seems quite standard. However, there are some features with unusual data, most notably 'cholesterol', which has a remarkable number of zero values. According to what I have learned, zero serum cholesterol is unrealistic because cholesterol is a lipid molecule essential for various physiological processes, including the formation of cell membranes and the synthesis of hormones. Therefore, cholesterol serum equal 0 is unrealistic.

This could be due to a small mistake during data collection, but we cannot determine which value is correct to fix it. Therefore, I have decided to remove them from the dataset.

And here is what I got:

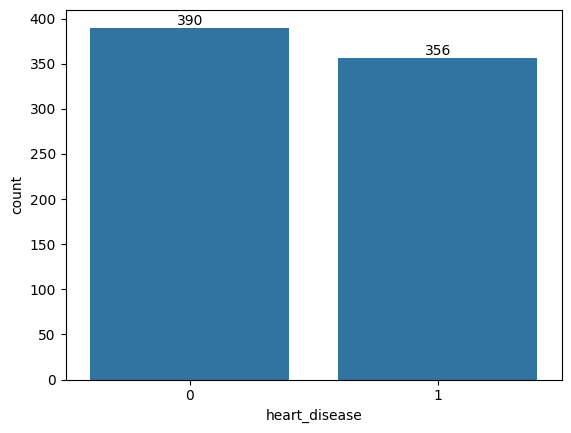


Figure 2‑4: Labels Countplot after cleaning

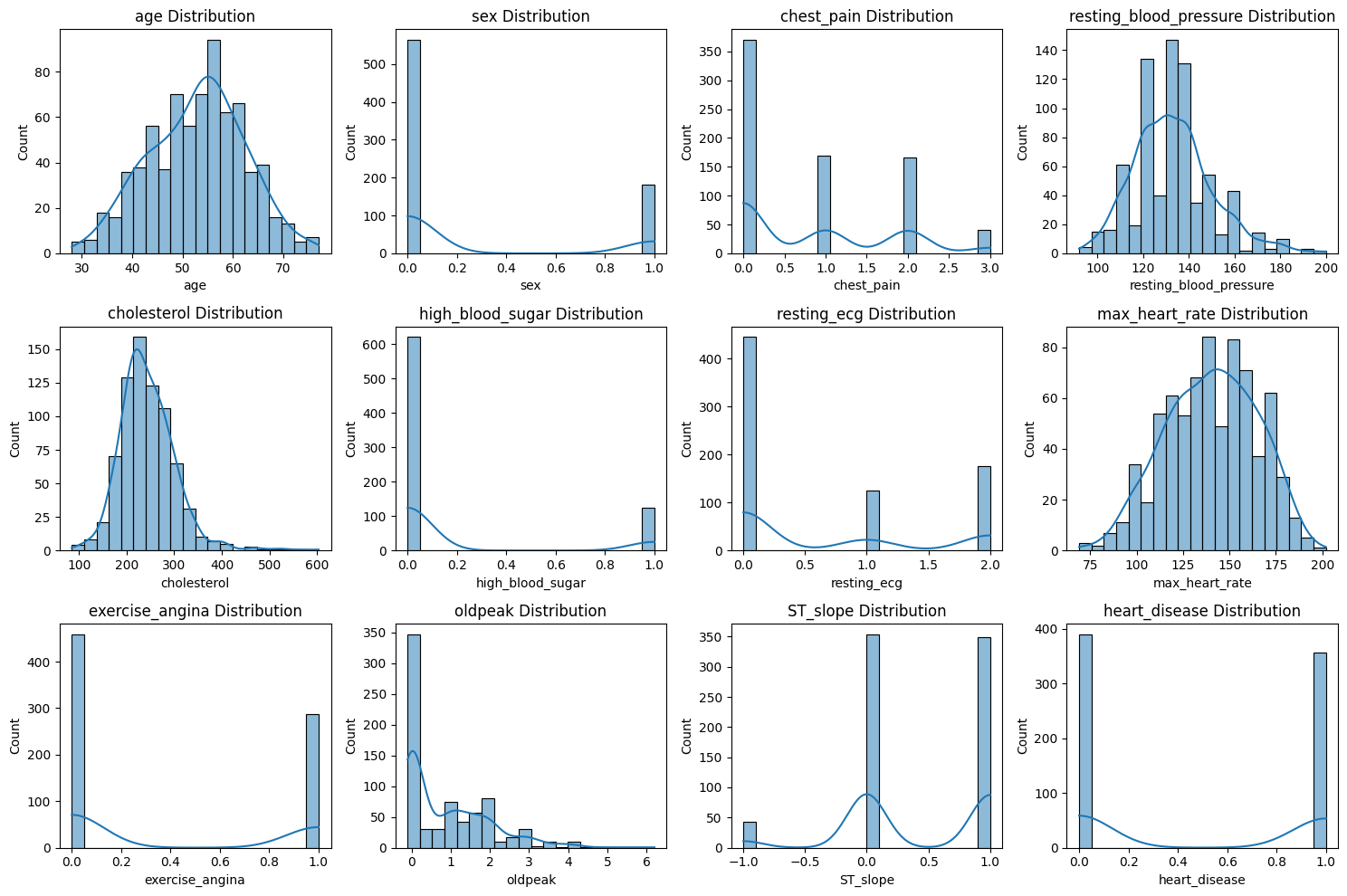


Figure 2‑5: Histplot Distributions after cleaning

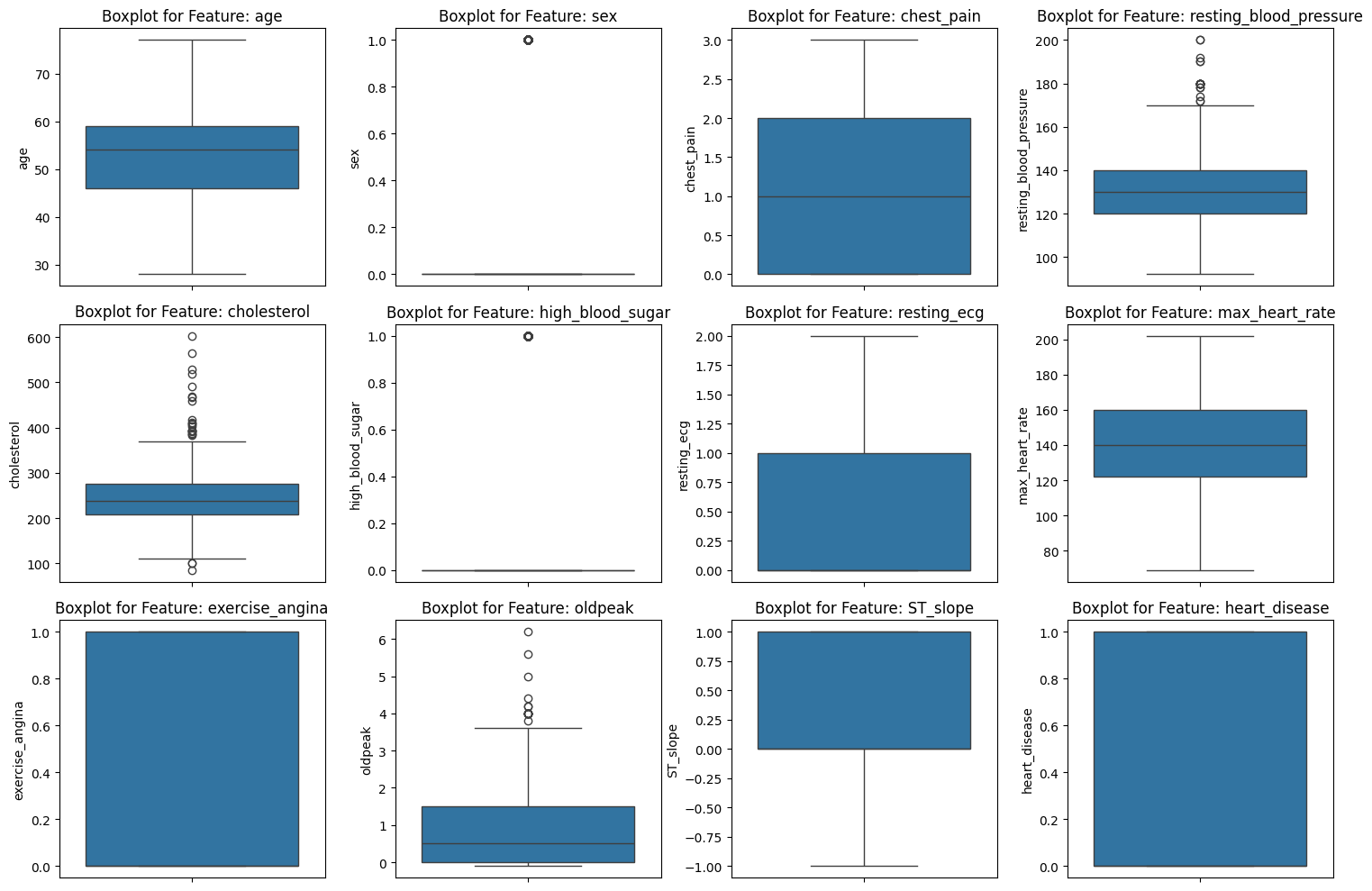


Figure 2‑6: Boxplot Distributions after cleaning

After removing approximately 172 instances with 'cholesterol' feature values equal to 0, I observed that 390 instances were labeled as 0 and 356 instances were labeled as 1, remaining in the dataset. The output labels appear to be fairly balanced, with nearly all feature distributions being standard.

Additionally, some very rare and unusual instances with extremely serious health problems signal, such as maximum heart rate less than 70 or resting blood pressure values equal to 0, have also been removed alongside those instances where 'cholesterol' feature values equal to 0. The distribution of all features is now more reliable.

## Analysis and Feature engineering

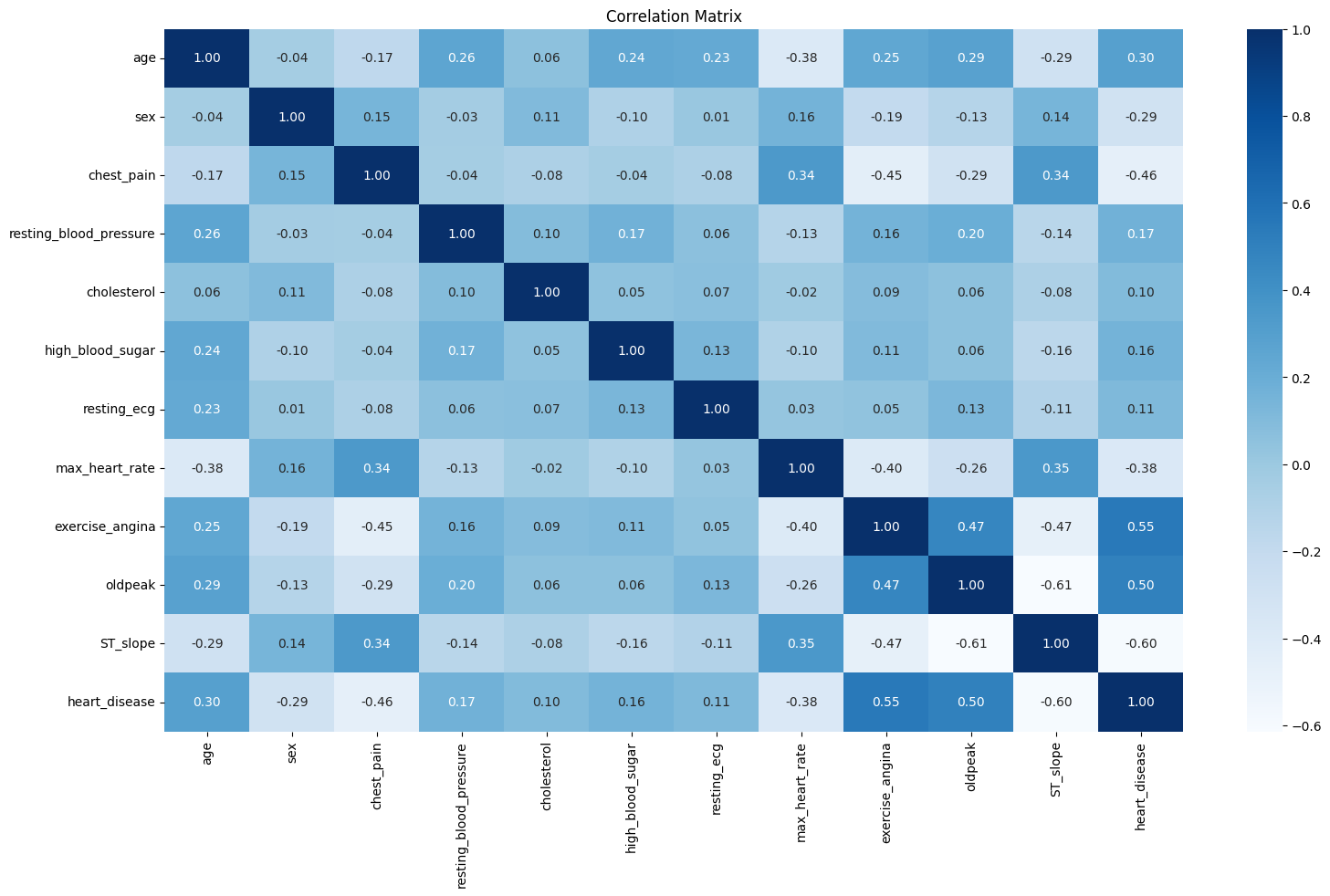


Figure 2‑7: Correlation Matrix

From the correlation matrix, we can observe that features such as 'age', 'exercise\_angina', and 'oldpeak' exhibit strong positive correlations with the output 'heart\_disease'. This suggests that higher values of 'age', 'exercise\_angina', and 'oldpeak' correspond to a higher likelihood of the output being 'heart\_disease' = 1. Conversely, features such as 'sex', 'chest\_pain', 'max\_heart\_rate', and 'ST\_slope' display strong negative correlations with the output 'heart\_disease'. This indicates that lower values of 'sex', 'chest\_pain', 'max\_heart\_rate', and 'ST\_slope' are associated with a higher likelihood of the output being 'heart\_disease' = 1. Other features have a slightly weaker correlation compared to the aforementioned ones.

For example, when examining the distribution of the 'age' feature:

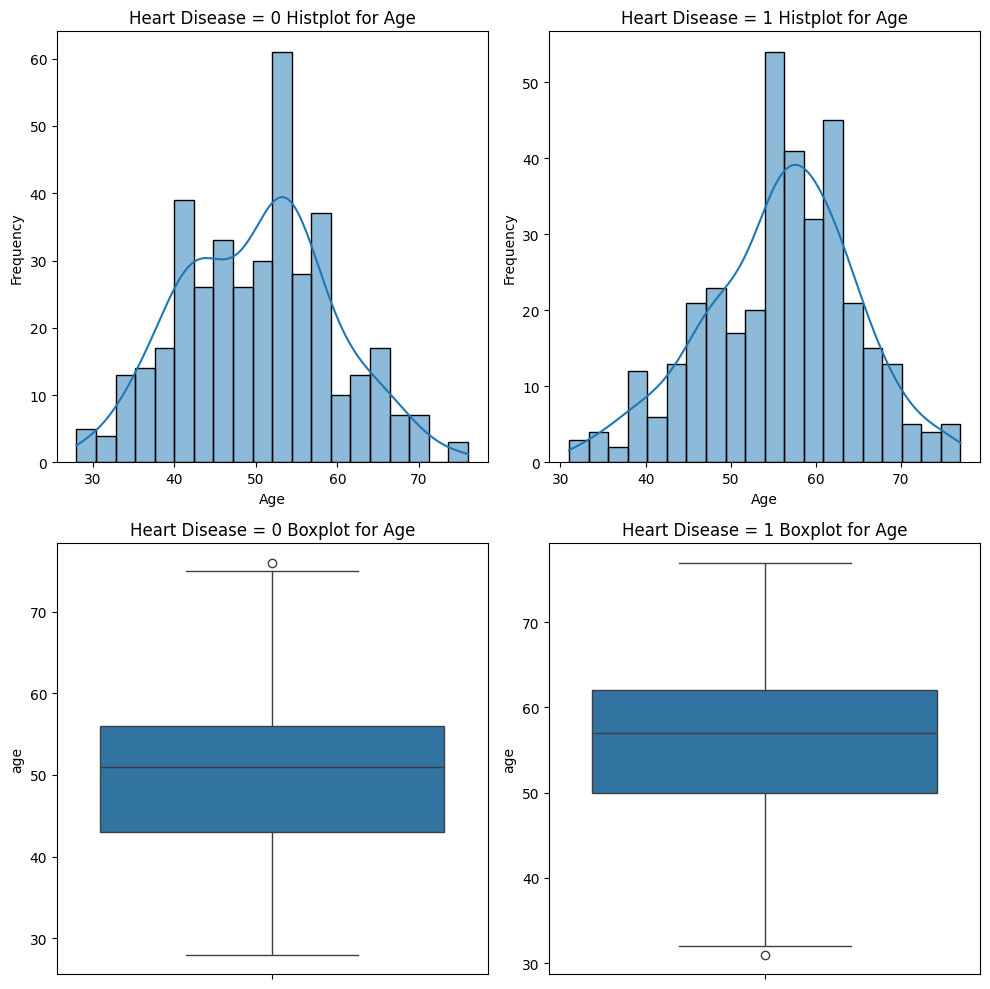


Figure 2‑8: Distribution of age feature by labels

The distribution of the 'age' feature differs between cases where the output is labeled as 0 and 1, particularly around the age of 54. When 'heart\_disease' = 0, there are numerous instances distributed in the age range lower than 54. Conversely, when 'heart\_disease' = 1, the distribution is more focused on the age range higher than 54.

Because of that, I have divided the 'age' feature into two ranges: lower than 54 and higher than 54, and engineered a new feature based on these two ranges.

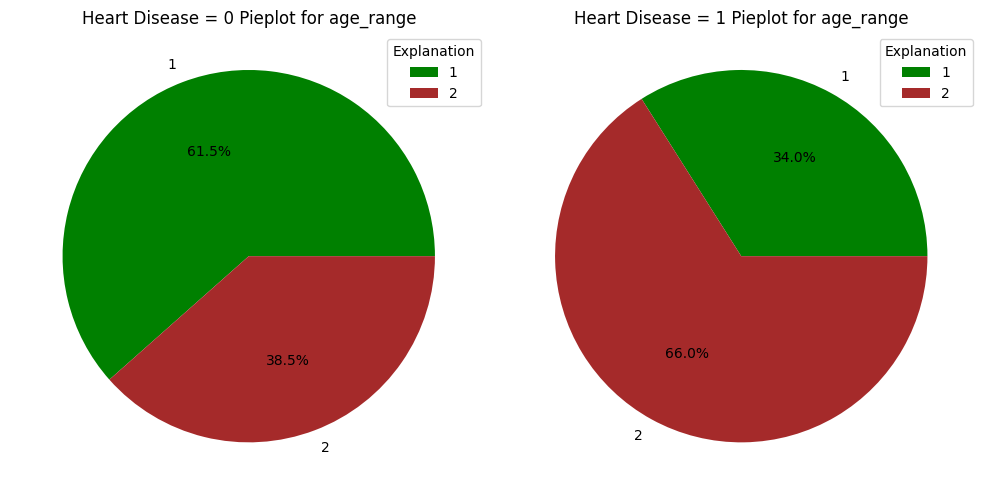


Figure 2‑9: Pieplot for age\_ranges

We can see the difference more clearly through these pie plots. When 'heart\_disease' value equals 0, the distribution of these instances focuses mostly on the lower range, below 54, represented by the value of 1. Conversely, when 'heart\_disease' value equals 1, the distribution of these instances focuses mostly on the higher range, above 54, represented by the value of 2.

As for the 'sex' feature

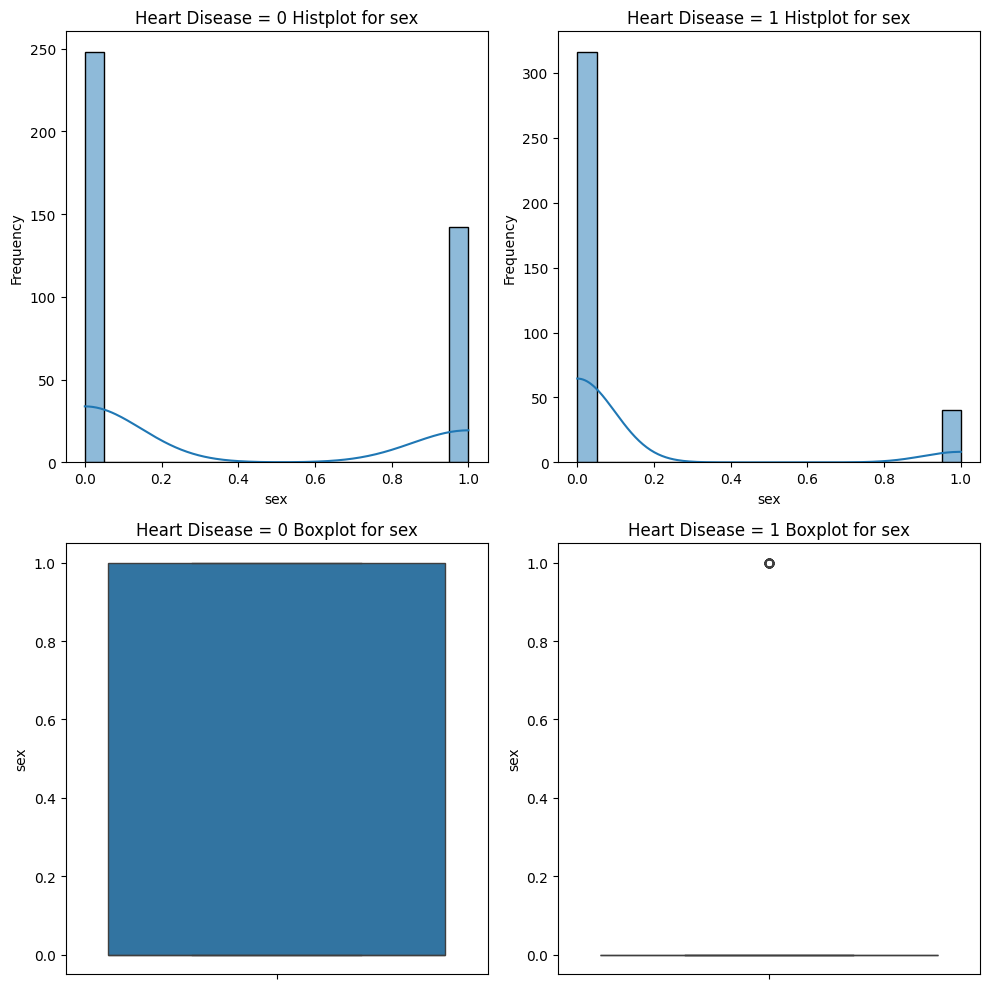


Figure 2‑10: Distribution of sex feature by labels

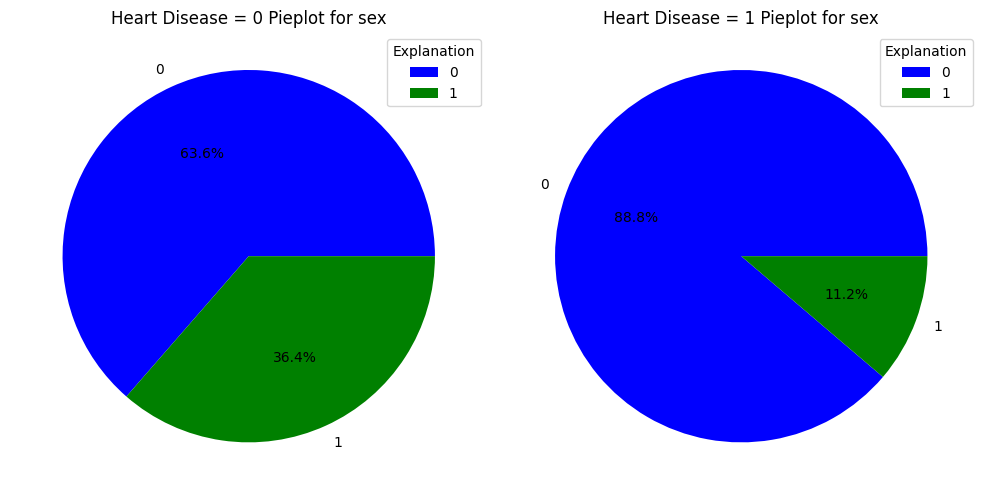


Figure 2‑11: Pieplot for sex

The distribution of instances when 'heart\_disease' = 1 and when 'heart\_disease' = 0 is quite similar. The proportion of sex = 0 (male) is always larger than sex = 1 (female), but there is a slight difference between them. Specifically, the proportion of 'sex' = 1 when 'heart\_disease' = 0 is much higher than when 'heart\_disease' = 1, indicating that among all heart disease patients, males have more instances than females. I assume that males have a higher chance of having heart disease than females.

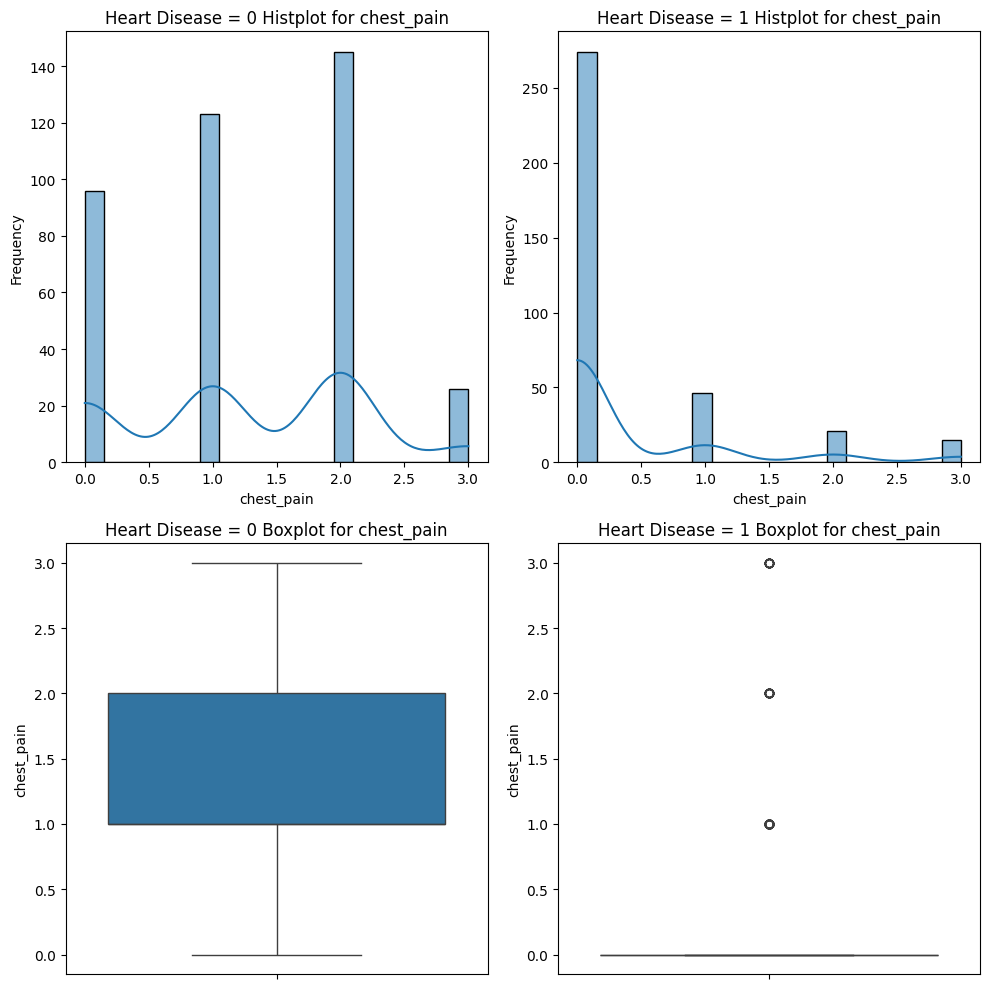


Figure 2‑12: Distribution of chest\_pain feature by labels

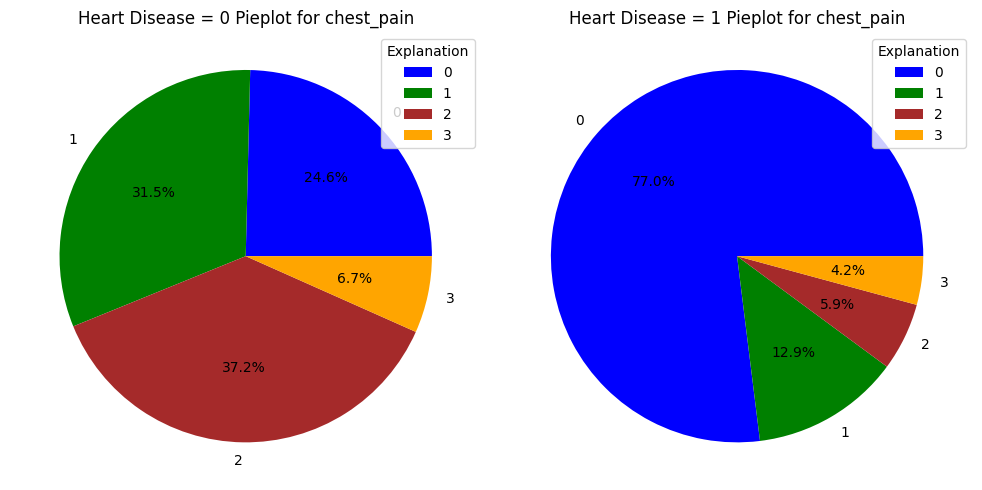


Figure 2‑13: Pieplot for chest\_pain

The distribution of the 'chest\_pain' feature differs significantly between instances where 'heart\_disease' = 1 and 'heart\_disease' = 0. When 'heart\_disease' = 0, the distribution appears more evenly spread across all values of the 'chest\_pain' feature, where 0 representing asymptomatic, 1 representing non-anginal pain, 2 representing atypical angina, and 3 representing typical angina. In contrast, when 'heart\_disease' = 1, the majority of instances focus on 'chest\_pain' = 0, indicating that most instances with heart disease are asymptomatic. This suggests that the instances collected may predominantly represent minor illness cases.

For features with weak correlations, for example 'resting\_blood\_pressure' and 'cholesterol' feature.

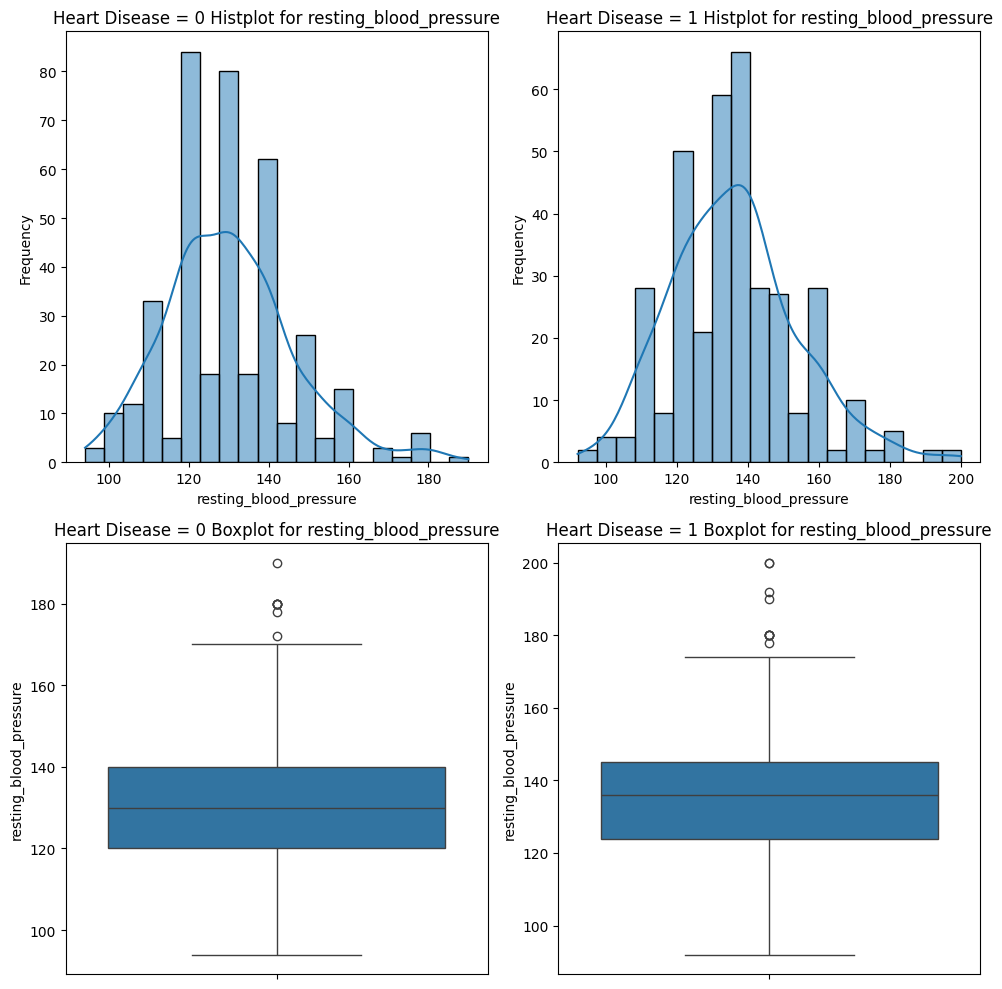


Figure 2‑14: Distribution of resting\_blood\_pressure feature by labels

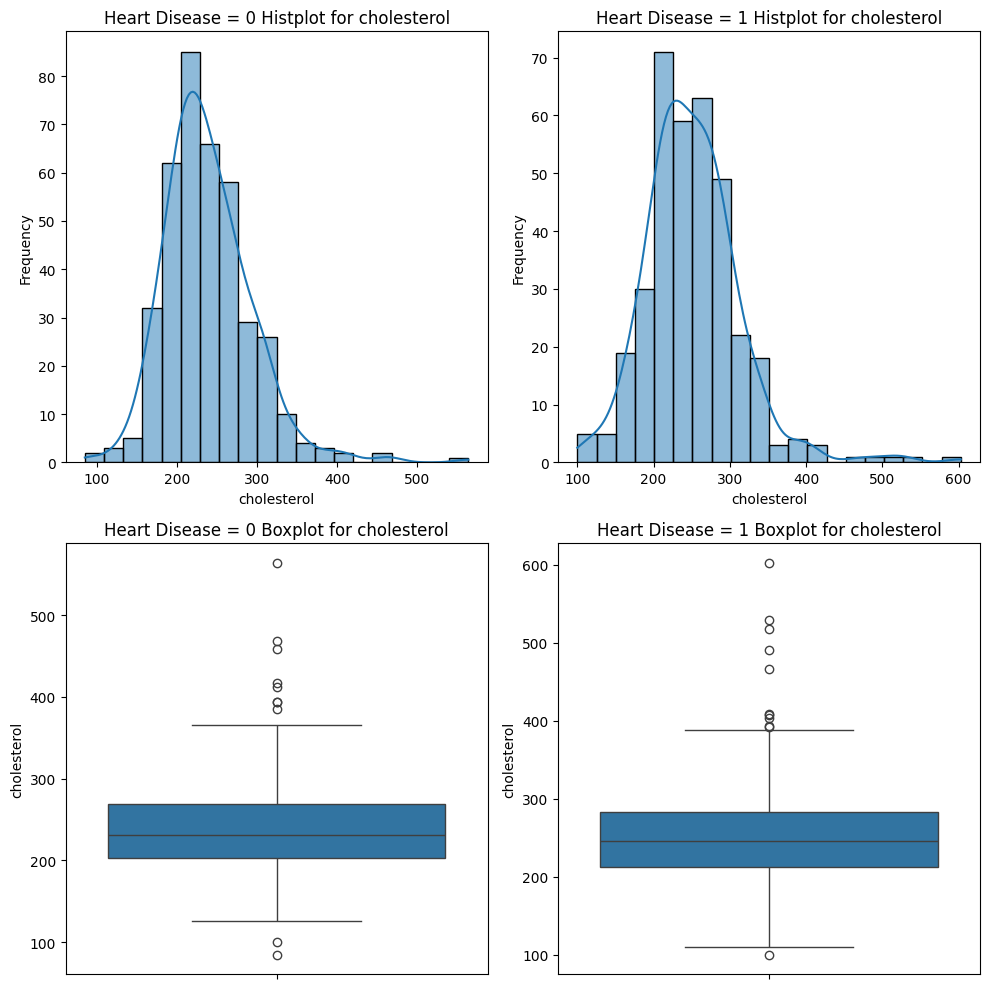


Figure 2‑15: Distribution of cholesterol feature by labels

We can observe similar distributions in both cases where 'heart\_disease' = 0 and 'heart\_disease' = 1. The main difference is the proportion of each feature value in each case.

Particularly in the cases of 'high\_blood\_sugar' and 'resting\_ecg'

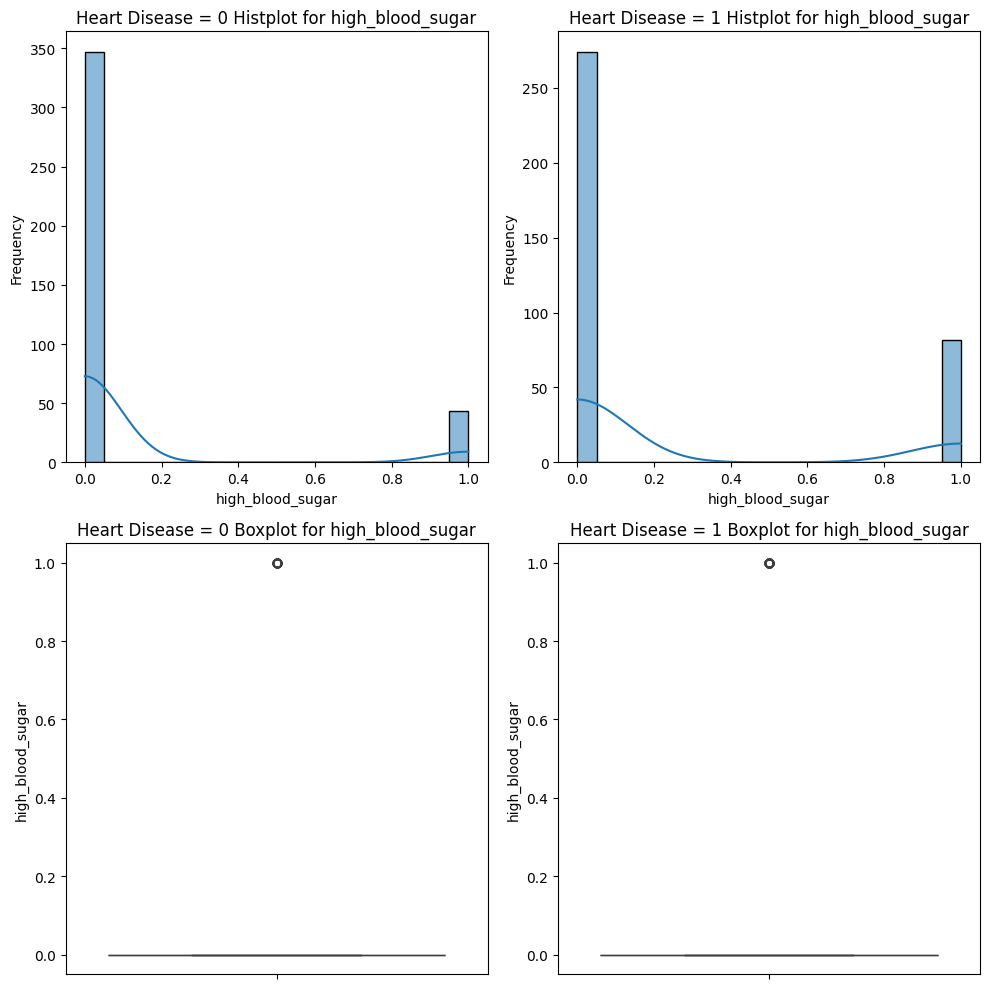


Figure 2‑16: Distribution of high\_blood\_sugar feature by labels

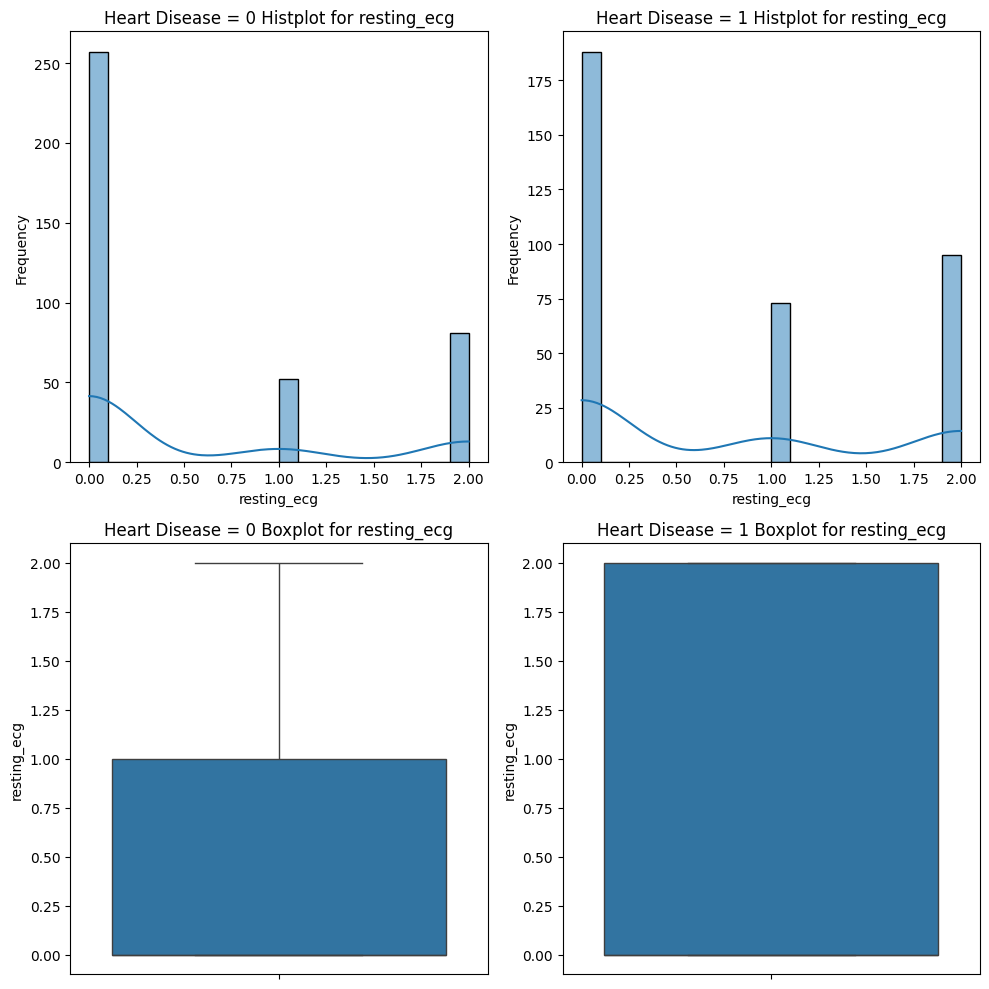


Figure 2‑17: Distribution of resting\_ecg feature by labels

The distributions resemble that of the 'sex' feature case, but the differences are not large enough to be of concern, unlike in the 'sex' feature case.

In the case of the 'max\_heart\_rate' feature

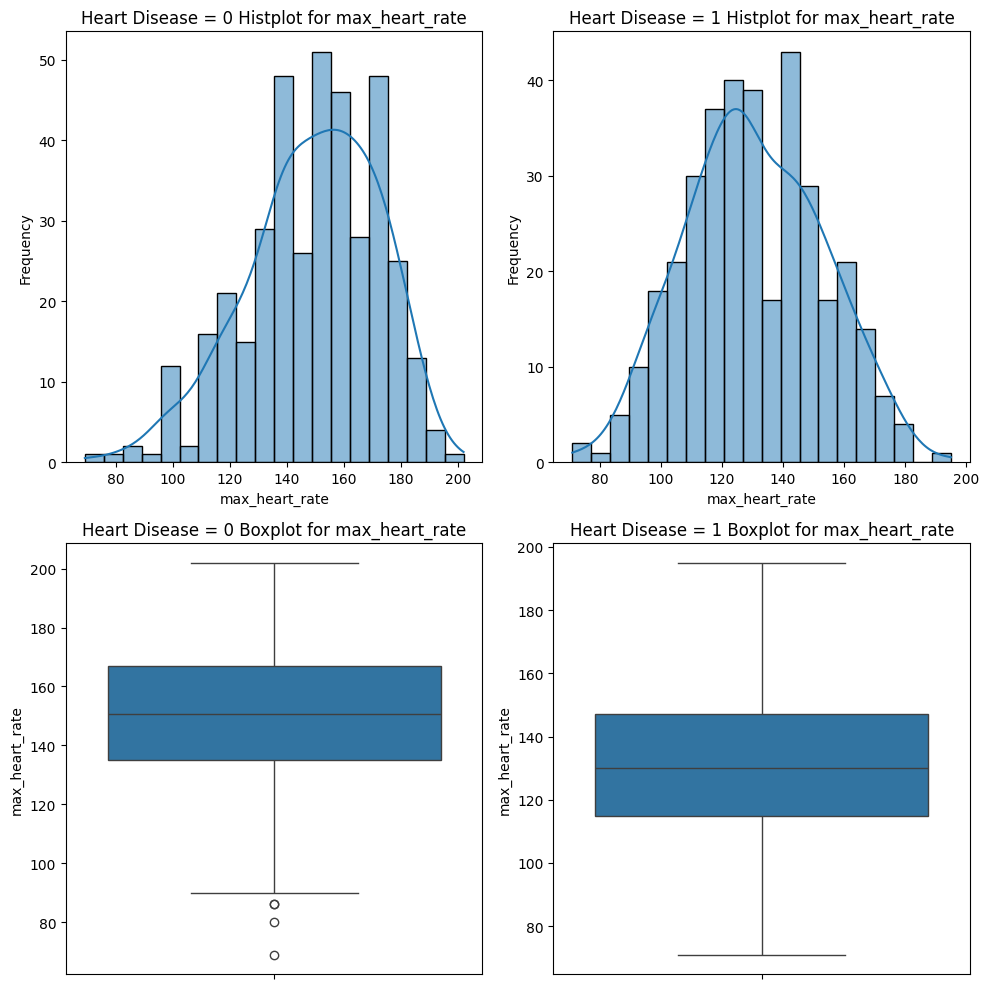


Figure 2‑18: Distribution of max\_heart\_rate feature by labels

It is similar to the 'age' feature in that its distribution is clearly divided into two ranges for the two different 'heart\_disease' labels. However, it is opposite to the 'age' feature in that when 'heart\_disease' = 0, most instances focus on a range higher than 140. Conversely, when 'heart\_disease' = 1, a significant number of instances focus on a range lower than 140.

So I engineered a new feature by dividing the 'max\_heart\_rate' feature into two ranges: one for values lower than 140 presented by 1 and another for values higher than 140 presented by 2.

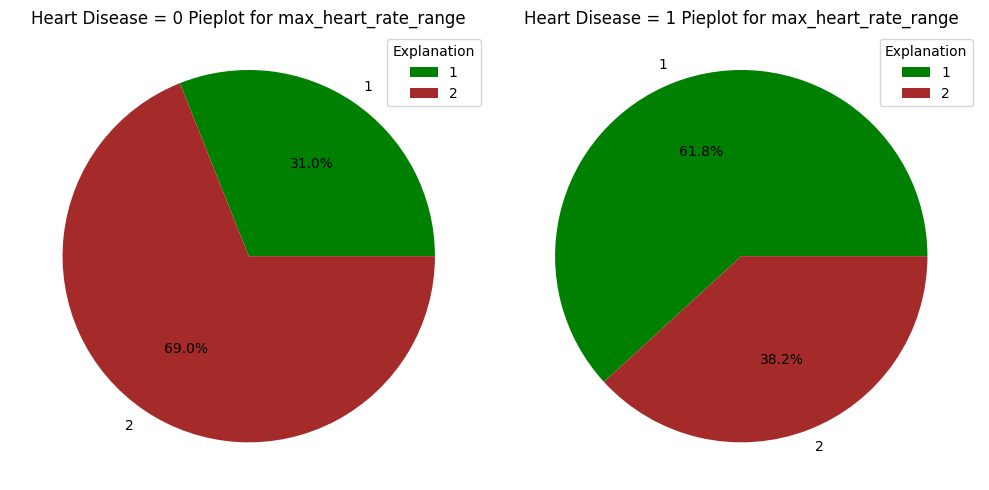


Figure 2‑19: Pieplot for max\_heart\_rate\_range

The pie plots show us a clearer difference in distribution between both cases of 'heart\_disease' from the two ranges of maximum heart rate values, which I named 'max\_heart\_rate\_range'.

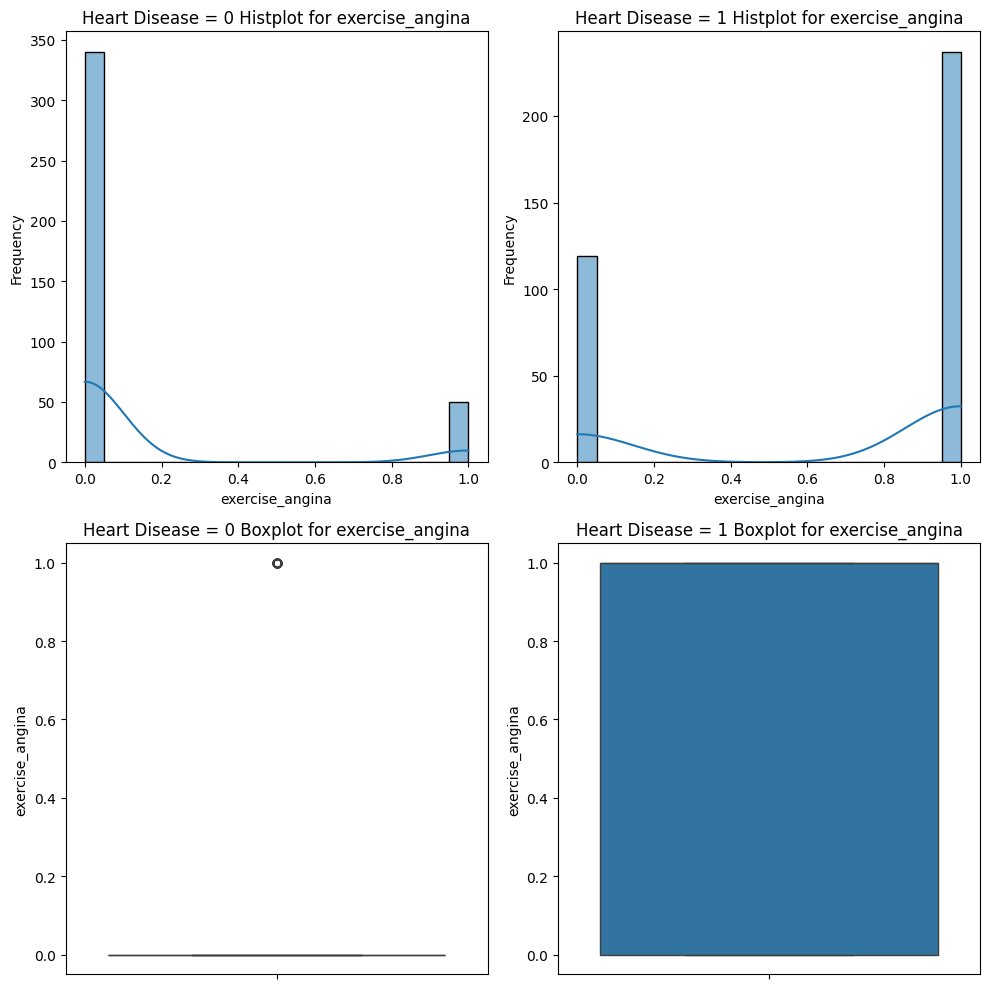


Figure 2‑20: Distribution of exercise\_angina feature by labels

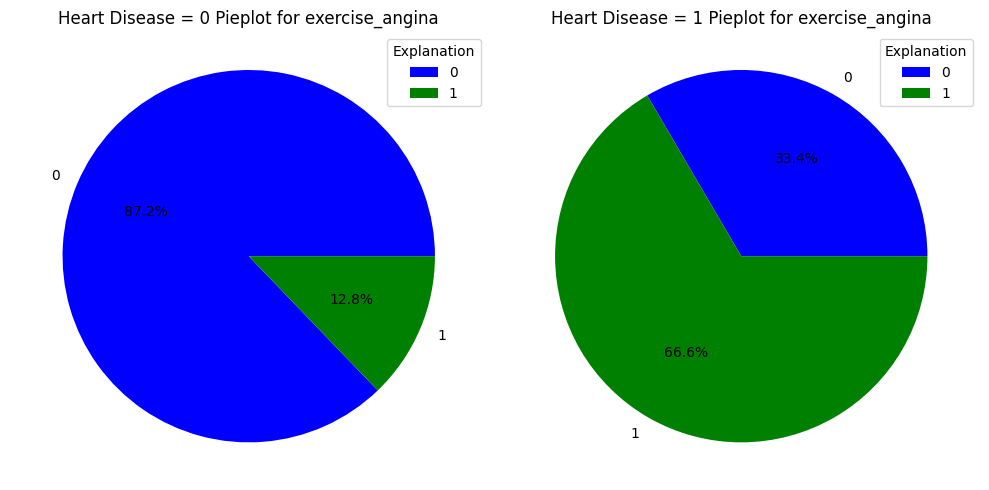


Figure 2‑21: Pieplot for exercise\_angina

The distribution of the 'exercise\_angina' feature shows a significant difference between instances where 'heart\_disease' = 0 and 'heart\_disease' = 1. Specifically, 66.6% of instances with heart disease exhibit exercise angina, while 87.2% of instances without heart disease do not show any problems with exercise angina. Showing that this 'exercise\_angina' feature is very important.

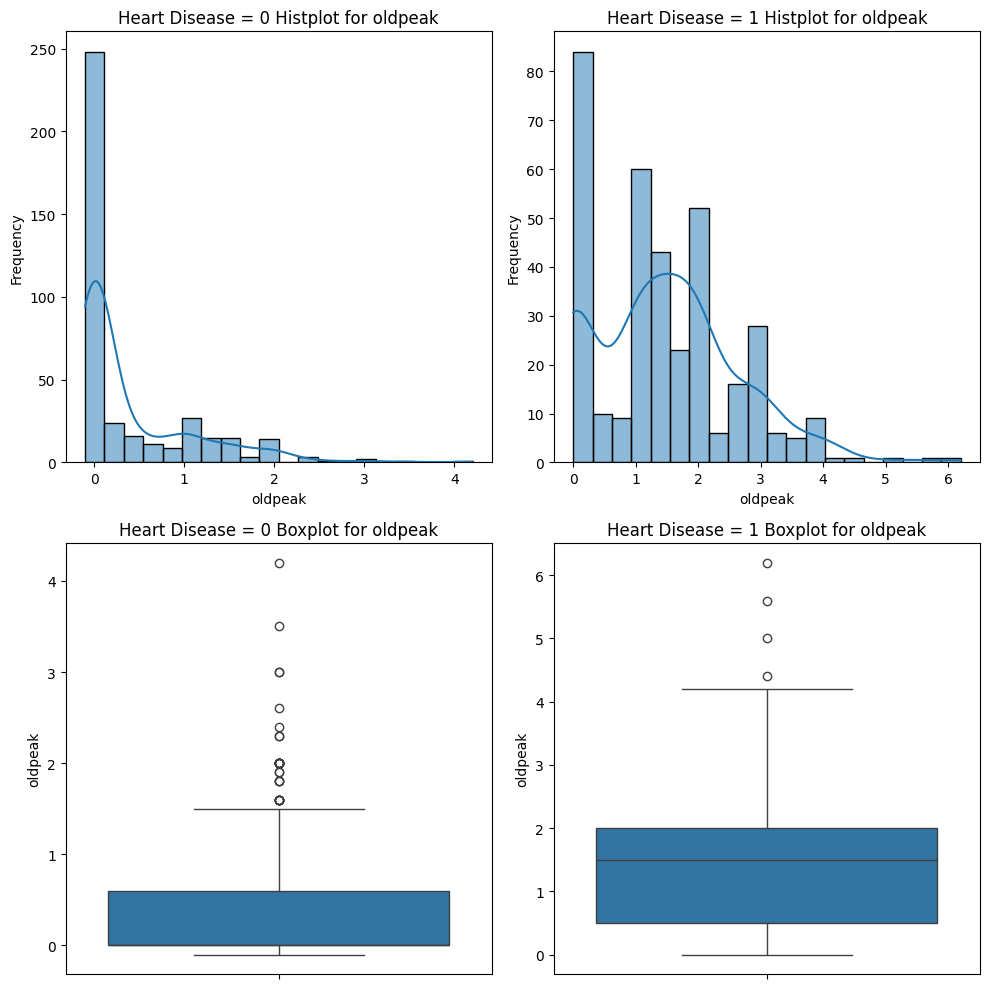


Figure 2‑22: Distribution of oldpeak feature by labels

In the medical context, 'oldpeak' typically refers to the ST depression induced by exercise relative to rest. This parameter is often utilized in cardiac stress testing to evaluate the severity of coronary artery disease. Through the distribution of 'oldpeak', it becomes evident that most instances without heart disease tend to have an oldpeak value around 0. This suggests that if oldpeak is 0, it often implies that the patient's heart is likely to function normally during physical exertion, without any ischemia (lack of blood flow) or other abnormalities detected by the ST segment changes. However, in instances where 'heart\_disease' = 1, there are many instances with oldpeak values different from 1. In the context of this dataset, most of them are greater than 0, but based on my research, it could also be lesser than 0.

In the case of 'ST\_slope' distribution

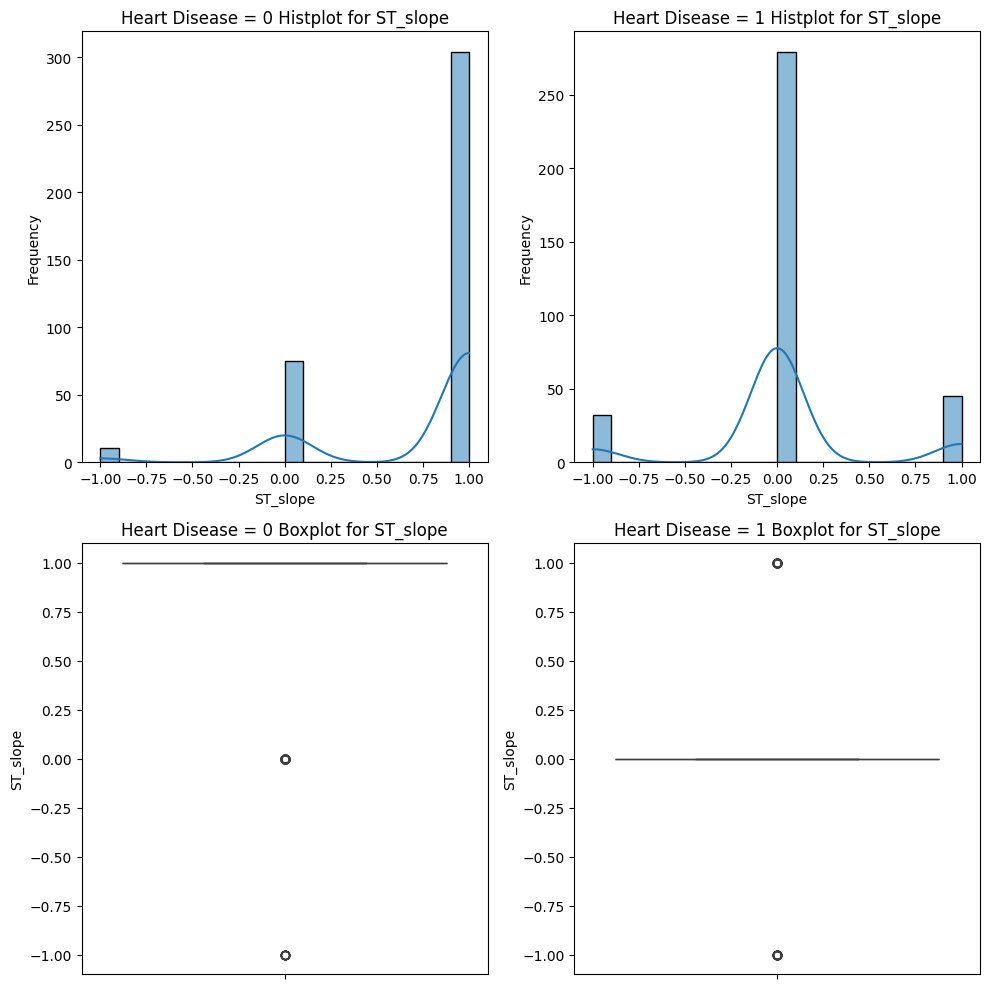


Figure 2‑23: Distribution of ST\_slope feature by labels

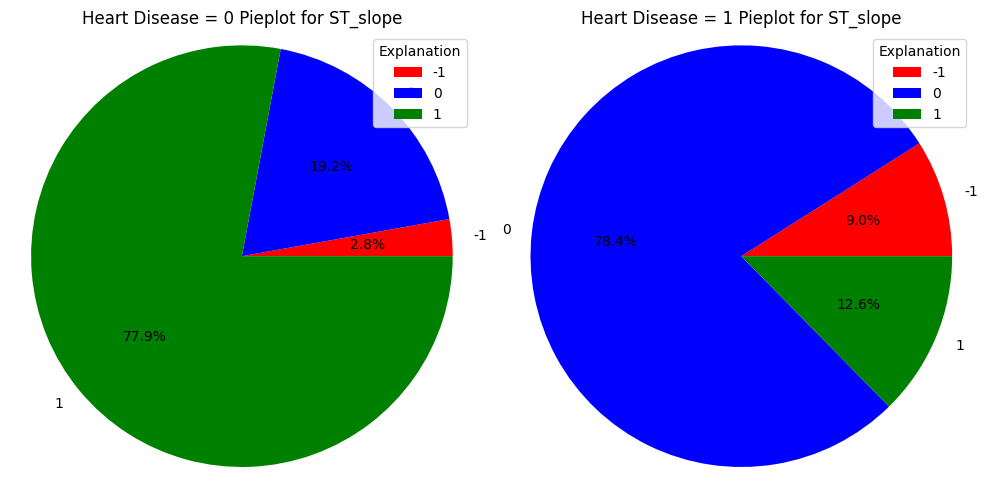


Figure 2‑24: Pieplot for ST\_slope

It is notable that most instances with 'heart\_disease' = 0 have an ST slope value of 1, indicating upsloping of the ST segment. Conversely, in instances where 'heart\_disease' = 1, the majority focus on an ST slope value of 0, representing a flat ST segment. Additionally, instances with an ST slope value of -1, indicating downsloping of the ST segment, have a higher number of occurrences compared to instances where 'heart\_disease' = 0. This underscores the importance of the 'ST\_slope' feature in predicting heart disease.

After preprocessing and feature engineering during the analysis phase, I removed unrealistic instances and engineered two additional features to enhance the dataset. Furthermore, duplicate instances were dropped, resulting in a refined dataset comprising 746 usable instances and a total of 13 features. Notably, features such as 'chest\_pain', 'max\_heart\_rate', 'exercise\_angina', 'oldpeak', and 'ST\_slope' demonstrated higher scores in the correlation matrix, indicating their potential importance compared to other features.

# Methodology

Within the scope of this project, I will use Logistic Regression, SVM, Random Forest, and k-Nearest Neighbors models for testing because it is a classification problem.

## Logistic Regression

Logistic Regression is a supervised machine learning algorithm used for binary classification tasks. Despite its name containing "regression," it's actually a classification algorithm commonly used when the target variable is categorical.

Logistic Regression is specifically used for binary classification problems where the target variable has two possible outcomes (e.g., yes/no, true/false, 0/1).

The algorithm uses the sigmoid function to map predictions between 0 and 1, representing probabilities. This function ensures that the output is in the range [0, 1], making it suitable for classification.

Logistic Regression finds a decision boundary in the input feature space to separate data points into different classes based on their features.

How It Works:

* Hypothesis Function: The logistic regression model calculates the weighted sum of input features along with a bias term.
* Sigmoid Activation: It then applies the sigmoid (logistic) function to the output of the hypothesis function. The sigmoid function maps any real-valued number to the range [0, 1].
* Loss Function: Logistic Regression uses a loss function called log loss (or cross-entropy loss) to measure the error between predicted and actual values.
* Optimization: The objective is to minimize this loss function by adjusting the model parameters (weights and bias) using optimization techniques like Gradient Descent.

Logistic Regression is a fundamental algorithm in machine learning and serves as a starting point for many classification problems. However, it's limited to linear decision boundaries and might not perform well on complex datasets. In such cases, more sophisticated algorithms like Support Vector Machines (SVMs) or ensemble methods like Random Forests or Gradient Boosting might be more suitable.

## Support Vector Machines (SVM)

Support Vector Machines (SVMs) are powerful supervised machine learning models used for classification and regression tasks. They are effective for both linear and non-linear data separation in classification and can also be used for regression.

SVMs are primarily used for classification tasks but can perform regression as well.

The main idea behind SVMs in classification is to find the optimal hyperplane that maximizes the margin between different classes. The hyperplane is the decision boundary that best separates different classes in the feature space.

The main idea behind SVMs in classification is to find the optimal hyperplane that maximizes the margin between different classes. The hyperplane is the decision boundary that best separates different classes in the feature space.

Support vectors are the data points closest to the hyperplane and play a crucial role in defining the decision boundary. These points influence the position and orientation of the hyperplane.

SVMs can efficiently handle non-linear boundaries by using a technique called the kernel trick. Kernels transform the input data into a higher-dimensional space, making it possible to find a linear separation in that space.

SVMs aim to maximize the margin between classes, which leads to better generalization and potentially higher accuracy. Different kernel functions (e.g., linear, polynomial, radial basis function (RBF)) can be used to handle linear and non-linear data distributions. SVMs are less prone to overfitting, especially in high-dimensional spaces, due to the margin concept.

How It Works:

* Data Mapping: For non-linear data, SVMs map the input data into a higher-dimensional space using a kernel function.
* Optimal Hyperplane: In this higher-dimensional space, SVM finds the hyperplane that maximizes the margin between classes while minimizing classification errors.
* Kernel Trick: The use of a kernel function allows SVMs to implicitly compute the dot product between input samples in the higher-dimensional space without explicitly transforming the data.
* Support Vectors: The final decision boundary is determined by the support vectors, which are the data points closest to the hyperplane.

SVMs are versatile and can handle both linear and non-linear problems, making them popular in various domains. However, they might not perform well on larger datasets due to their computational complexity and memory requirements. Additionally, selecting the right kernel and tuning hyperparameters are essential for achieving optimal performance.

## Random Forest Classification

Random Forest Classification is used for classification tasks, where the goal is to predict the class or category of an input based on its features.

Random Forest is an ensemble of decision trees, where each tree is trained on a random subset of the training data and makes independent predictions.

The final prediction is determined by a majority vote (for classification) or averaging (for regression) across all the individual trees.

Key aspects of the Random Forest Classifier:

* Ensemble of Decision Trees: Random Forest builds a collection of decision trees during the training phase. Each decision tree is trained on a random subset of the training data, and each tree has the potential to make different predictions.
* Random Feature Selection: At each split in a decision tree, a random subset of features is considered. This introduces diversity among the trees and helps prevent overfitting.
* Bootstrap Aggregating (Bagging): Random Forest uses a technique called bootstrapping, where multiple random samples (with replacement) are drawn from the training dataset. Each decision tree is trained on one of these bootstrap samples.
* Voting for Classification: In classification tasks, each tree in the forest predicts a class for a given input. The final prediction is determined by a majority vote among all the trees.
* Hyperparameters: Random Forest has hyperparameters that can be tuned for optimal performance, such as the number of trees (n\_estimators), the maximum depth of each tree (max\_depth), and the number of features considered at each split (max\_features).
* Out-of-Bag (OOB) Score: Random Forest can estimate its own generalization performance using out-of-bag samples. These are data points not included in the bootstrap sample used to train a particular tree, and they can be used for validation.
* Feature Importance: Random Forest can provide information about the importance of each feature in making predictions. This can be useful for feature selection and understanding the impact of different features on the model's performance.

## K-Nearest Neighbors

The k-Nearest Neighbors is a type of instance-based or lazy learning algorithm, where the model is not explicitly trained during the training phase.

It memorizes the training instances and makes predictions for new data points based on their proximity to known examples in the feature space.

How it works:

* Training Phase: During the training phase, the algorithm simply stores the training dataset.
* Prediction Phase: For a new data point, the algorithm identifies the k-nearest neighbors in the feature space. The prediction is then made based on the majority class (for classification) or the average (for regression) of the labels of the k-nearest neighbors.
* Distance Metric: The choice of distance metric (e.g., Euclidean distance, Manhattan distance) is crucial and depends on the nature of the data.
* Hyperparameter k: The hyperparameter "k" represents the number of neighbors to consider when making a prediction. The value of k is typically chosen based on cross-validation or other model evaluation techniques.
* Decision Boundary: k-NN does not explicitly learn a decision boundary. Instead, it classifies or predicts based on the local neighborhood of a data point.
* Scalability: k-NN can be computationally expensive, especially as the size of the dataset increases. Techniques like KD-trees or Ball trees are used to optimize the search for nearest neighbors.

You can adjust the value of n\_neighbors based on your specific requirements and the characteristics of your dataset. Additionally, it's important to scale the features when using k-NN, as the algorithm is sensitive to the scale of the input variables.

# Model Development

I use the 'heart\_disease' column to create the output (y) and all the other columns to create the features (X) of the models. Then, I use the train\_test\_split() function to split it into a training set and a validation set. After that, I use the StandardScaler to rescale both the training set and validation set. Finally, I use them to train models, and I obtained the results.

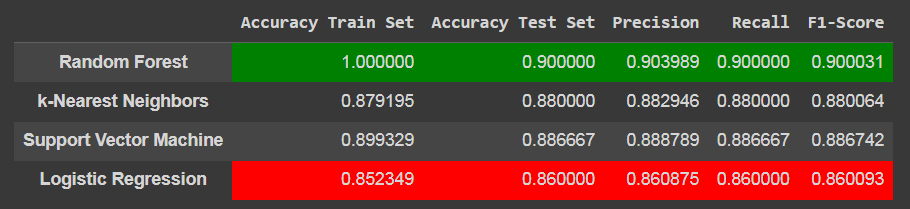


Figure 4‑1: Results of models

From the results, it is evident that the Random Forest model consistently achieves the highest scores in all evaluation metrics, followed by the SVM model. Conversely, the Logistic Regression model consistently produces the lowest scores across all evaluation metrics.

A Random Forest model achieving a training accuracy of 1.0 (or 100%) suggests a perfect fit to the training data. While this might seem impressive at first glance, it could also indicate potential issues, such as overfitting.

In disease prediction, prioritizing recall (sensitivity) to maximize the detection of true positive cases is typically crucial, followed by maintaining a balance between precision and recall (as indicated by the F1-score) to ensure accurate and reliable predictions. Given that the SVM model demonstrates the highest scores after Random Forest model in both metrics, it is the preferred choice for this dataset.

Finally, with GridSearchCV and the chosen hyperparameters, I tuned my model to achieve the following results:

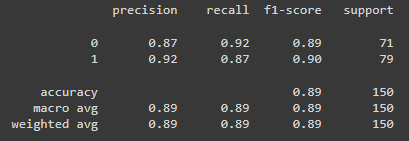


Figure 4‑2: SVM tuned results (1)

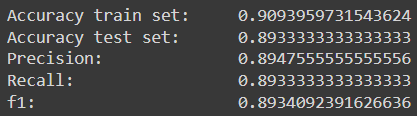
****

Figure 4‑3: SVM tuned results (2)

GridSearchCV improved the results of the SVM model, although not significantly. However, the outcome is now slightly better and closer to that of the Random Forest model, with no signs indicating potential issues such as overfitting.

# Conclusion

Overall, the dataset contains a plethora of information regarding heart disease. For example, among the common symptoms associated with heart disease, some show a higher association rate than others, such as chest pain, exercise-induced angina, oldpeak, and ST slope. People over 54 years of age and males show higher rates of heart disease than younger people and females. Additionally, individuals with a low maximum heart rate during vigorous exercise also demonstrate a higher likelihood of developing heart disease. These insights can inform advanced clinical diagnoses, as well as facilitate the implementation of early treatment and prevention methods to improve the condition before it worsens.

As for the model results, while the SVM model shows lower results than the Random Forest model in all metrics, the Random Forest model indicates potential issues, such as overfitting, when the training accuracy is 1.0, suggesting a perfect fit to the training data, which is much higher than the evaluation accuracy (0.9). After tuning with GridSearchCV, the result of the SVM model is now very close to that of the Random Forest model, with no signs indicating potential issues such as overfitting. Therefore, I believe the SVM model would be more suitable than the Random Forest model