Heart Disease Prediction

Preprocessing techniques:

1- **Dealing with empty cells:** done using imputation (replacing null values in numeric columns with the mean and replacing null values in categorical columns with the mode).

2- **Dealing with duplicate records:** checked for any duplicate records and none was found.

3- **Convert categorical columns to numerical columns:** done using one-hot encoding which assigns vectors to each category. The vector represents whether the corresponding feature is present (1) or not (0).

4- **feature scaling:** feature scaling is done using X_scaled = (x – x_min) / (x_max-xmin). this estimator scales and translates each feature individually such that it is in the given range on the training set, e.g., between zero and one.

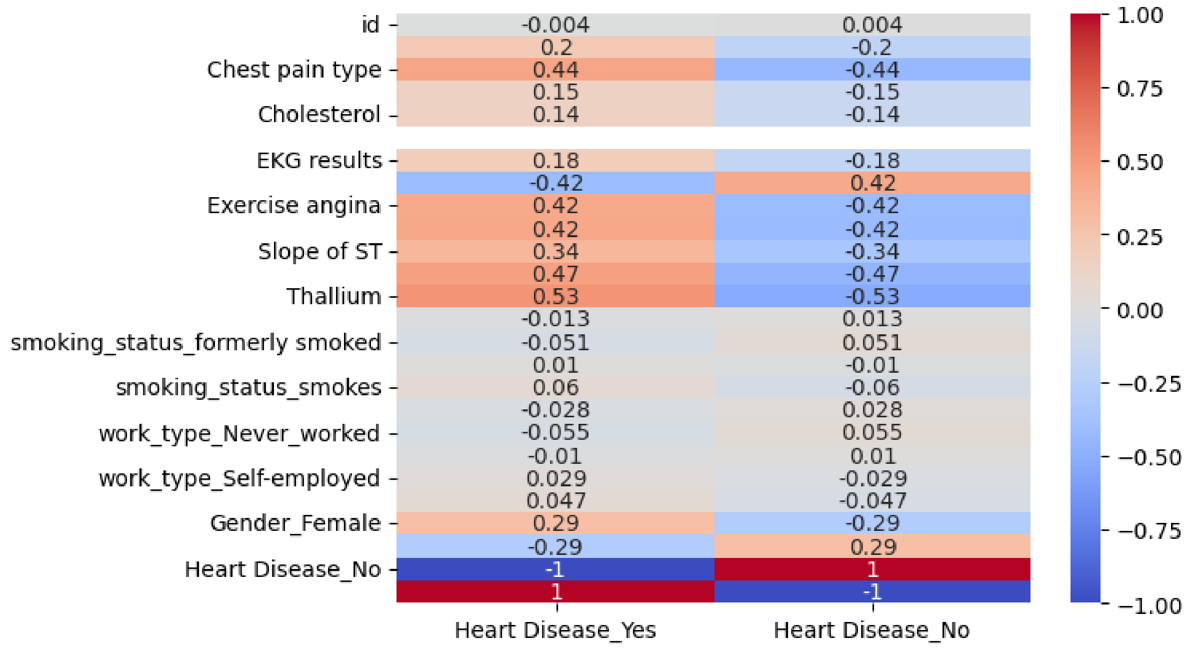
5- **Modifying data in wrong format:** checked for any wrong format data and none was found.

6- **Dealing with outliers:** after calculating the IQR outliers were replaced with the upper whisker if they are larger than it and replaced with the lower whisker if they are smaller than it.

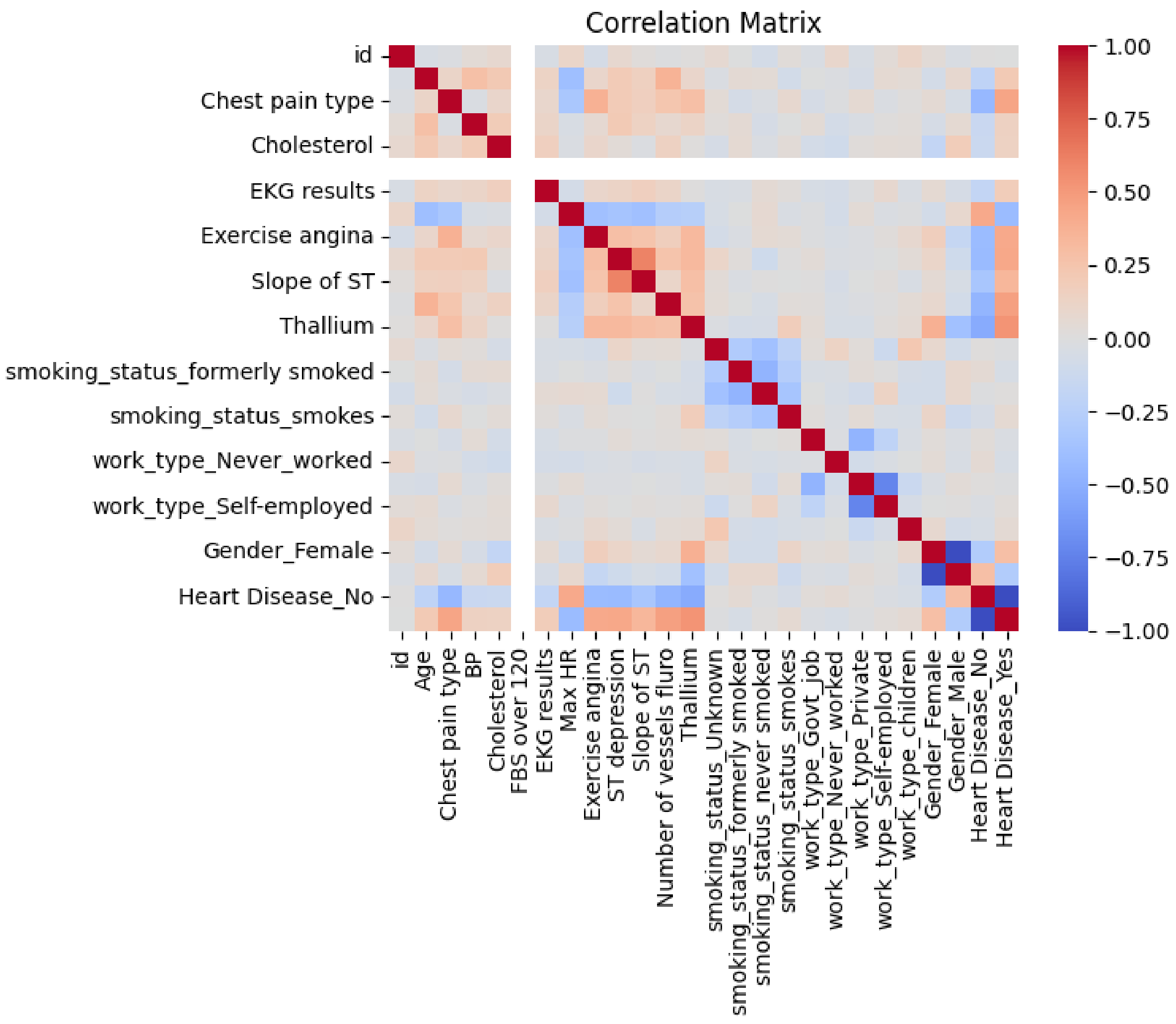
7- **Dealing with highly correlated features:** some features are highly correlated so one of them is removed as they are redundant (for example : Gender_Male and Gender_Female were higly correlated and Gender_Male was dropped).

Perform analysis and visualizations:

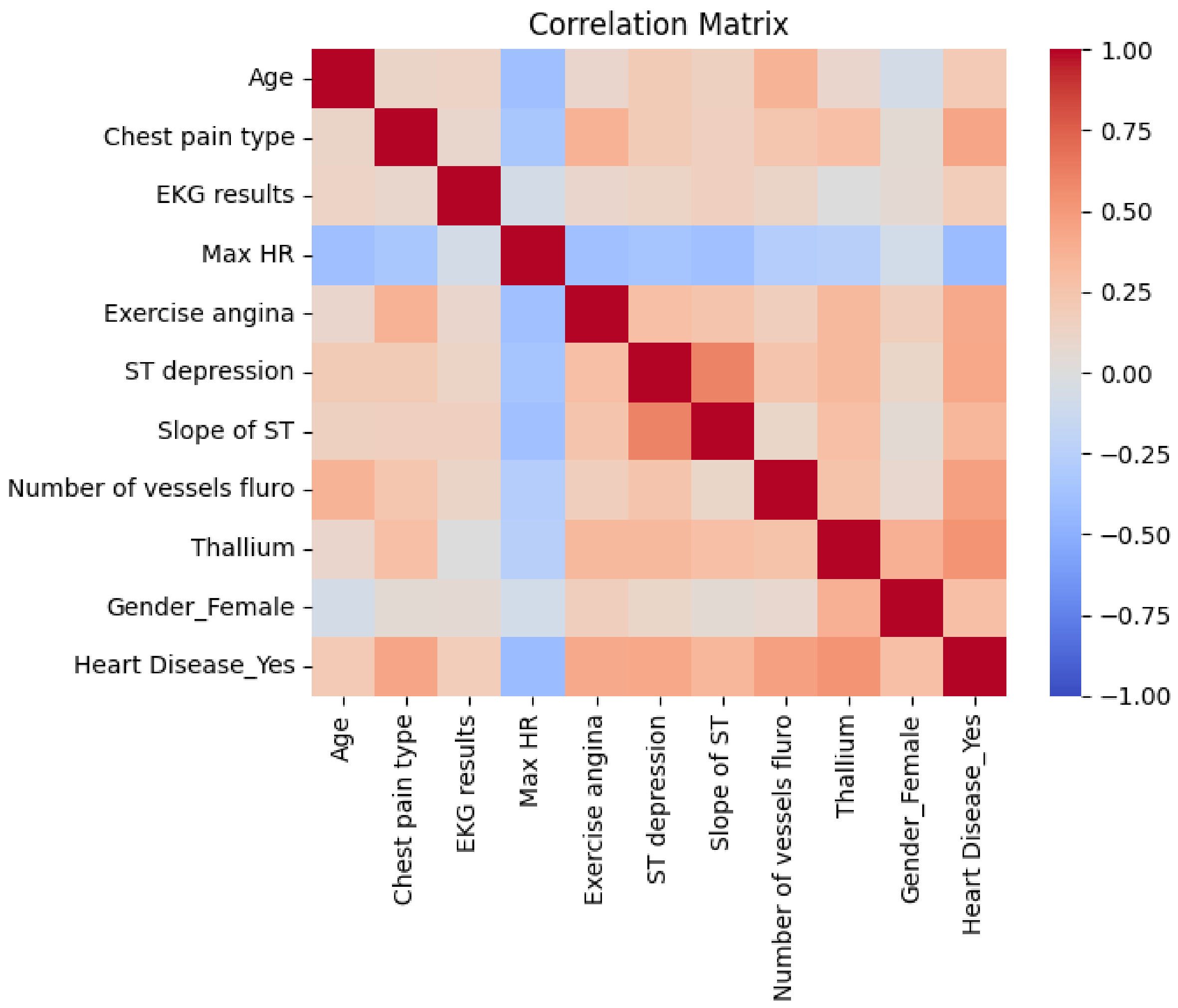
1- correlation between features and target variable



2- correlation matrix before dropping any columns



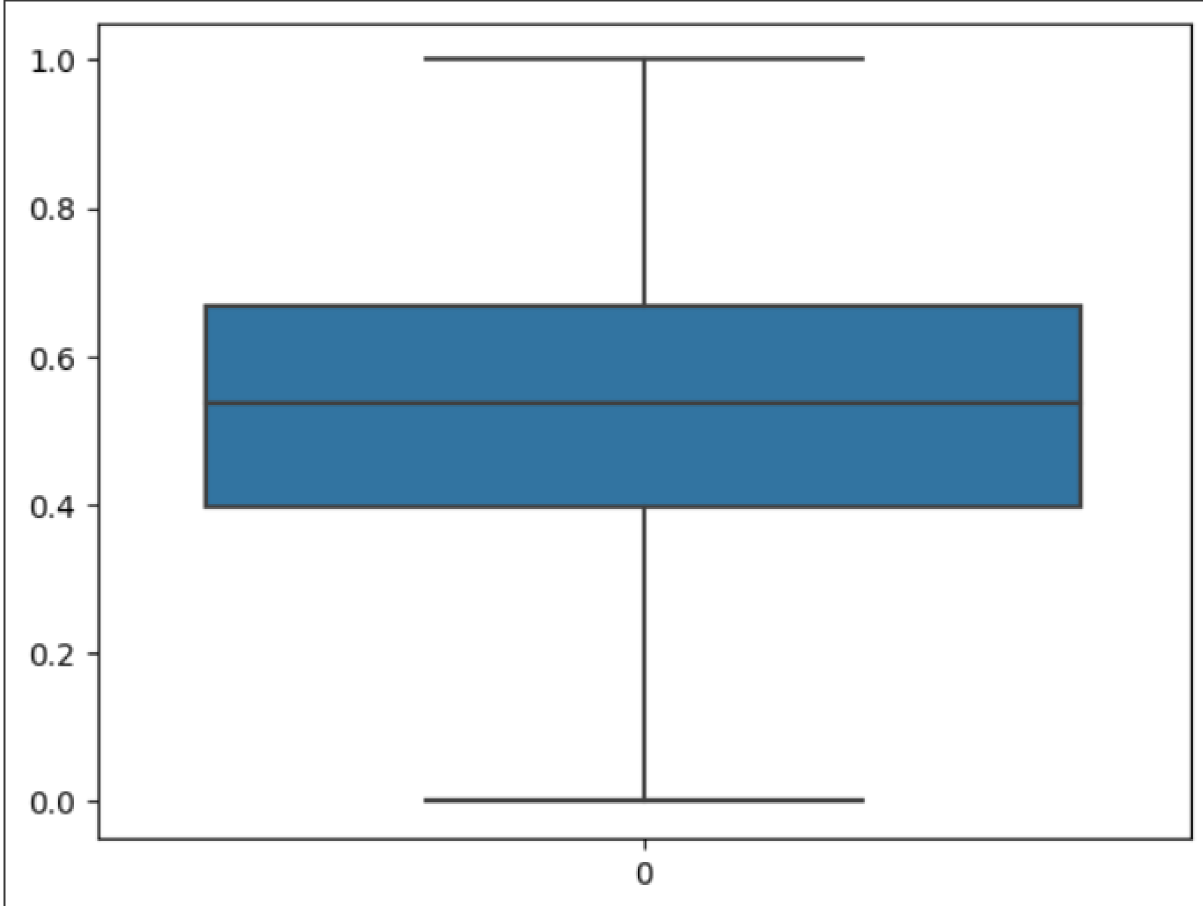
3- correlation matrix after dropping some columns



2- box plots used to visualize the outliers and data distribution

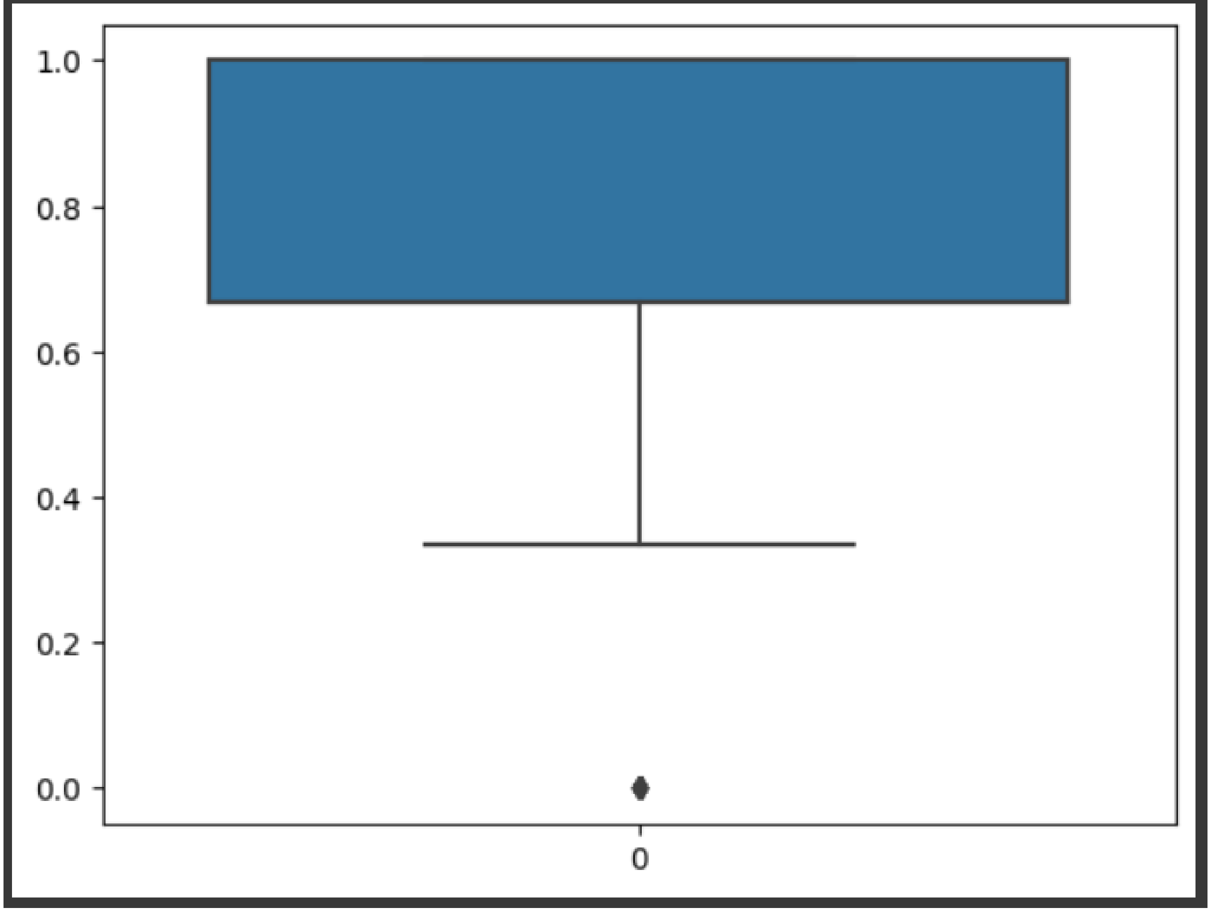
```
sns.boxplot(data['Age'])
```

```
<Axes: >
```



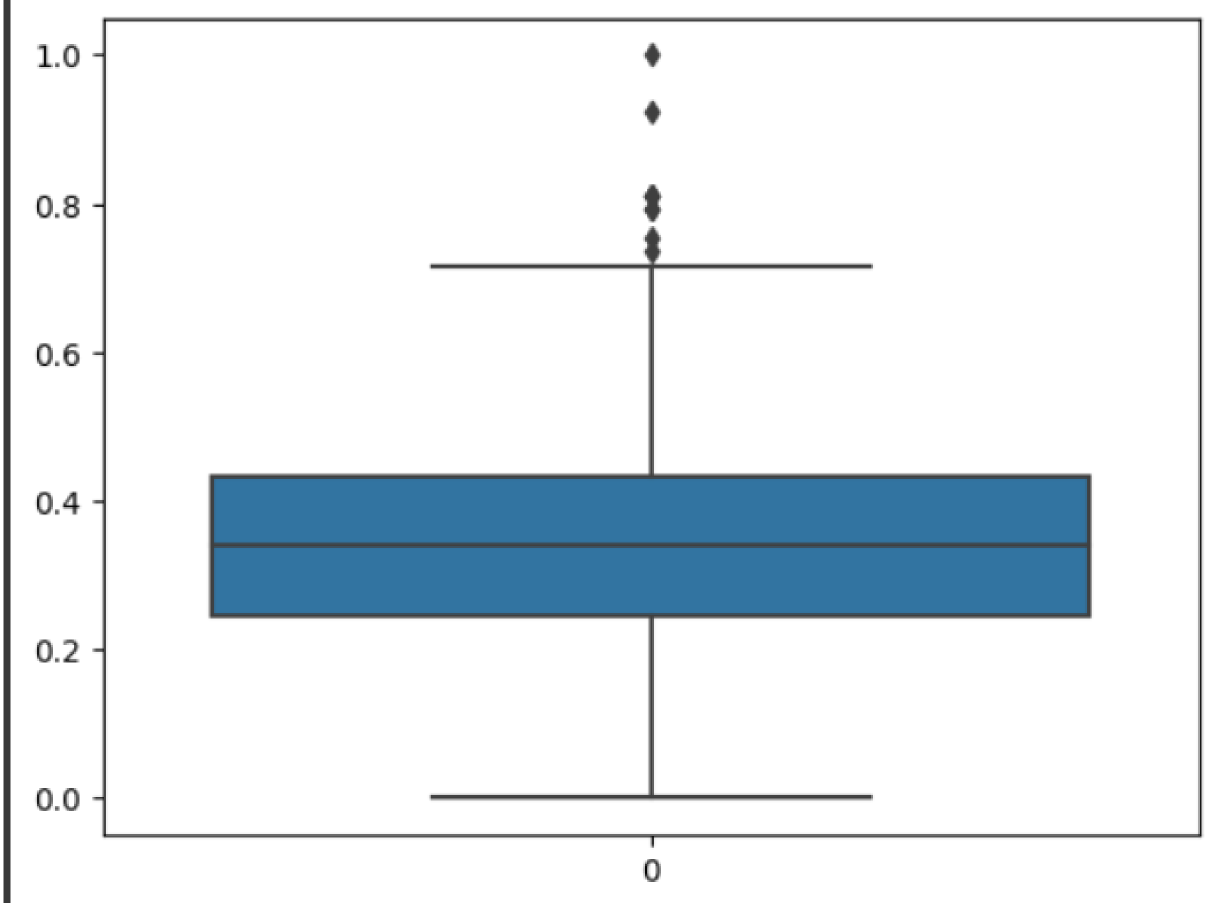
```
sns.boxplot(data['Chest pain type'])
```

```
<Axes: >
```



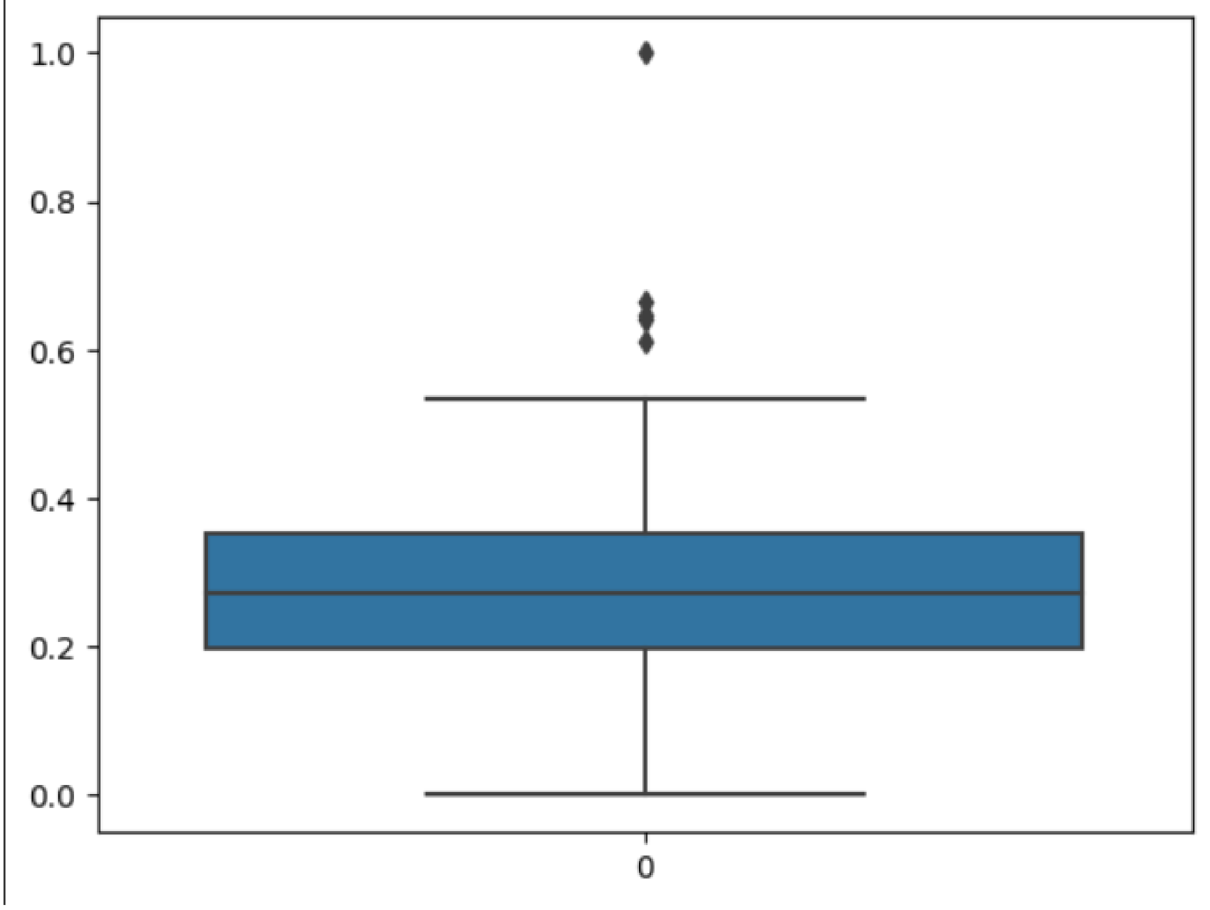
```
sns.boxplot(data['BP'])
```

```
<Axes: >
```



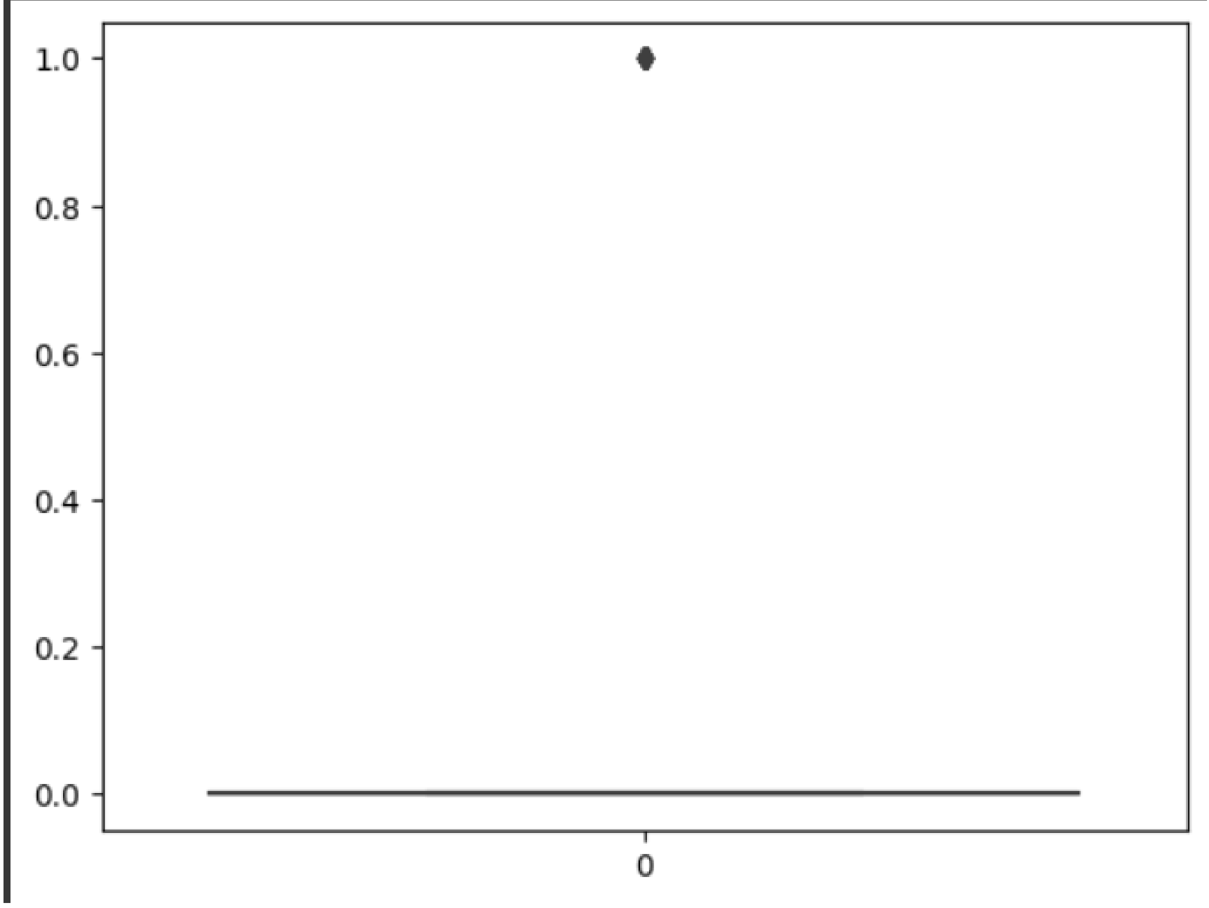
```
sns.boxplot(data['Cholesterol'])
```

```
<Axes: >
```



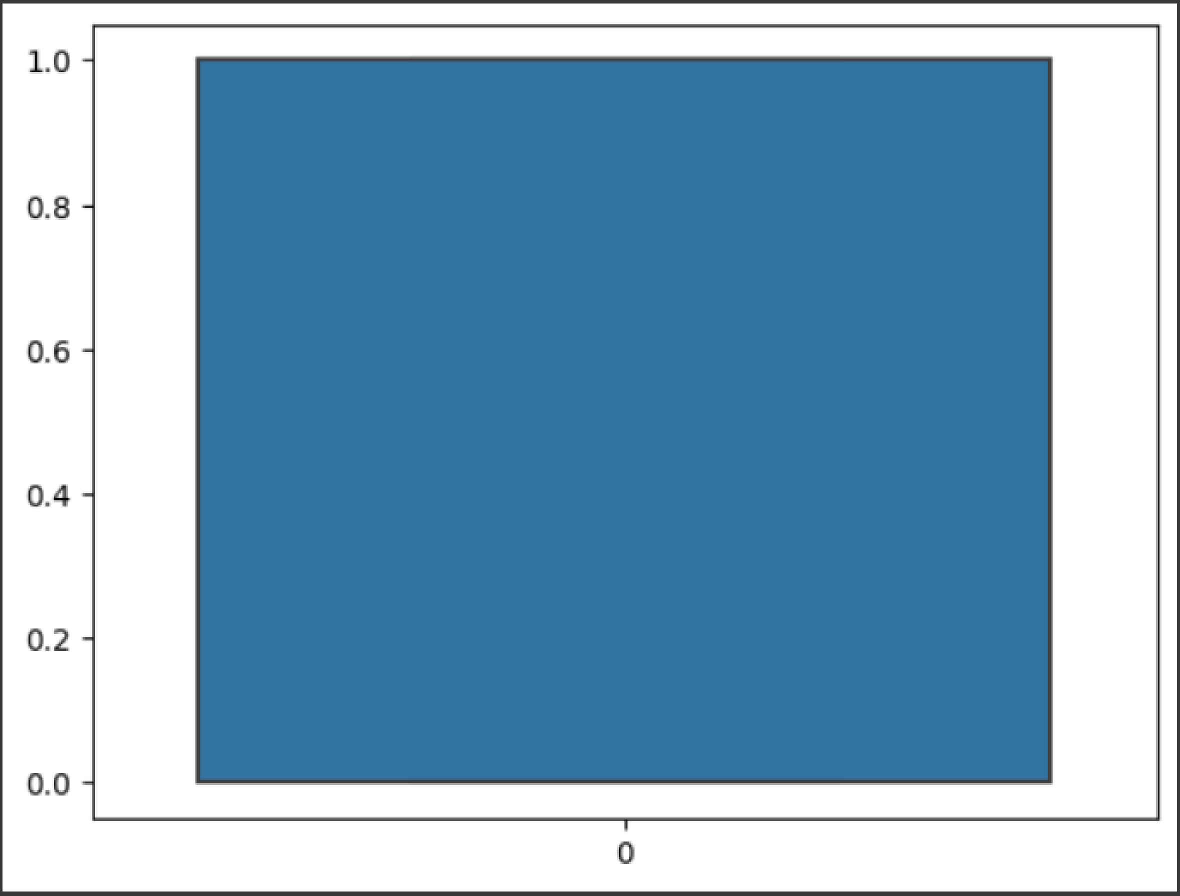
```
sns.boxplot(data['FBS over 120'])
```

```
<Axes: >
```



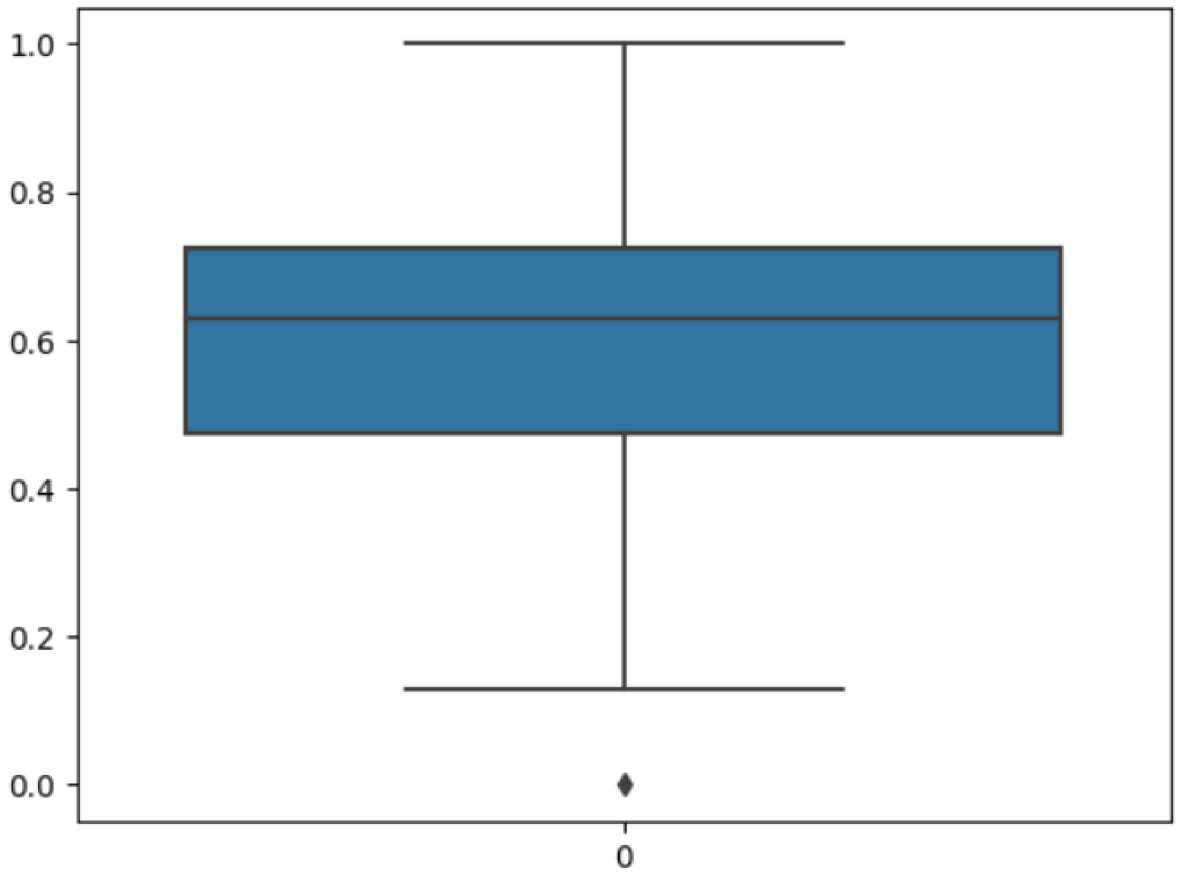
```
sns.boxplot(data['EKG results'])
```

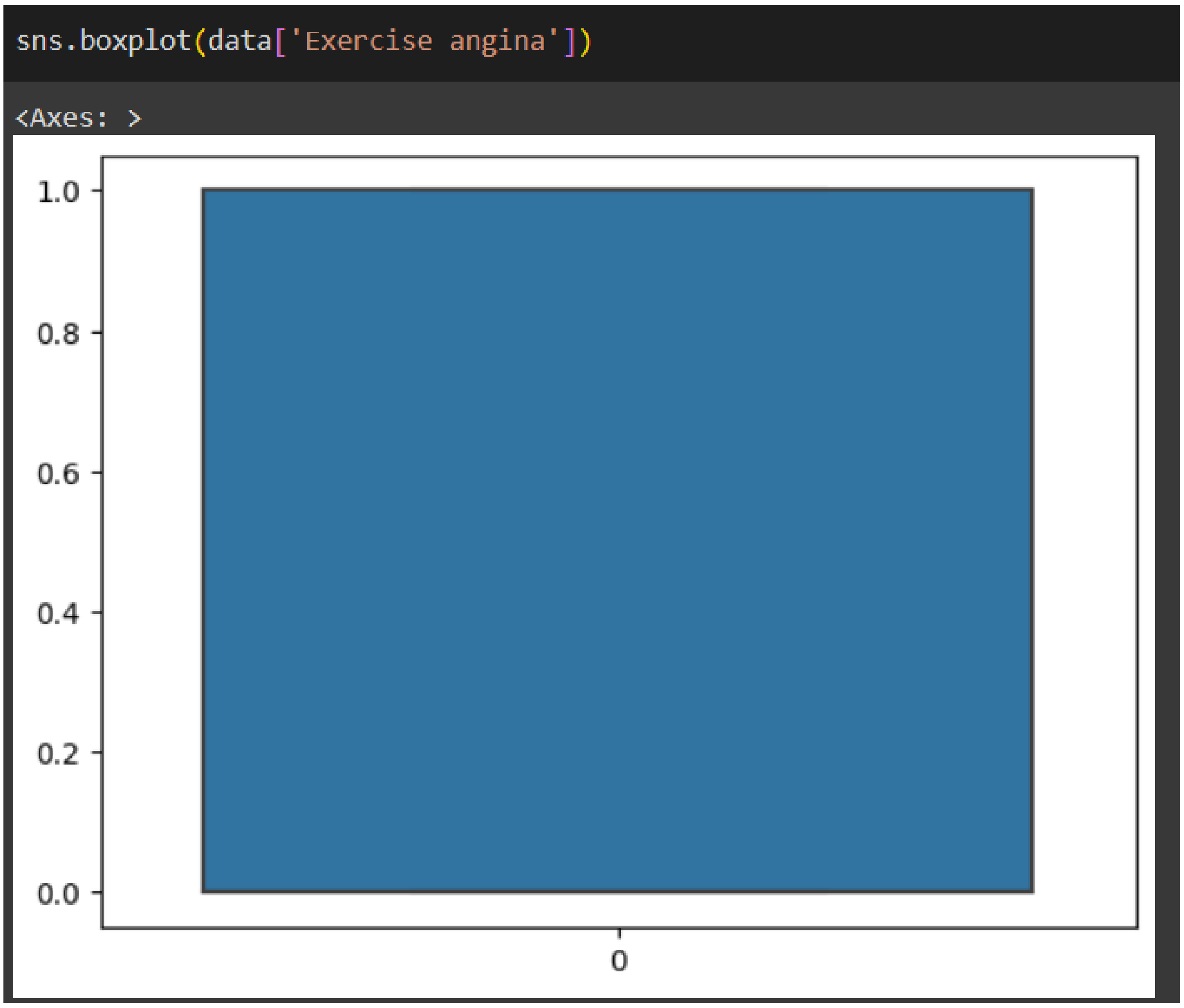
```
<Axes: >
```



```
sns.boxplot(data['Max HR'])
```

```
<Axes: >
```



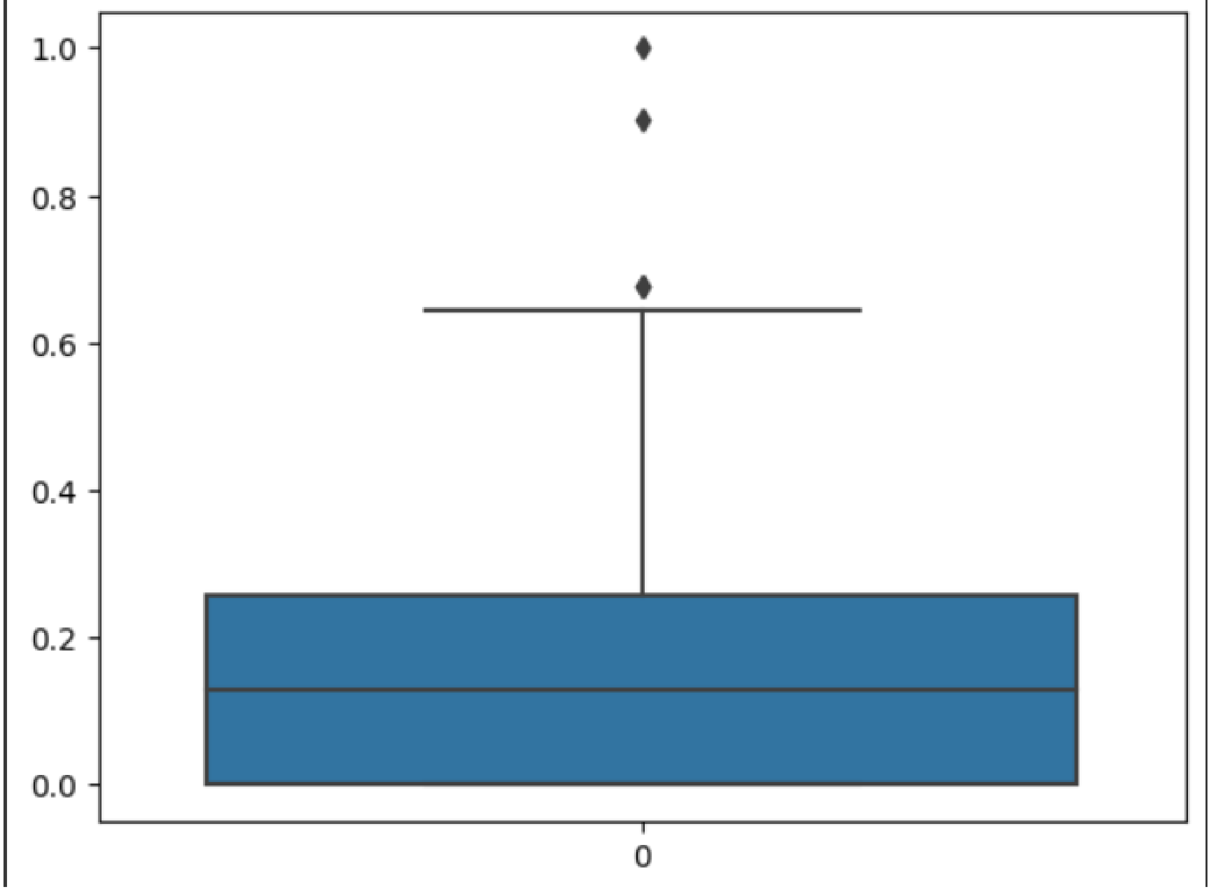


```
sns.boxplot(data['Exercise angina'])
```

```
<Axes: >
```

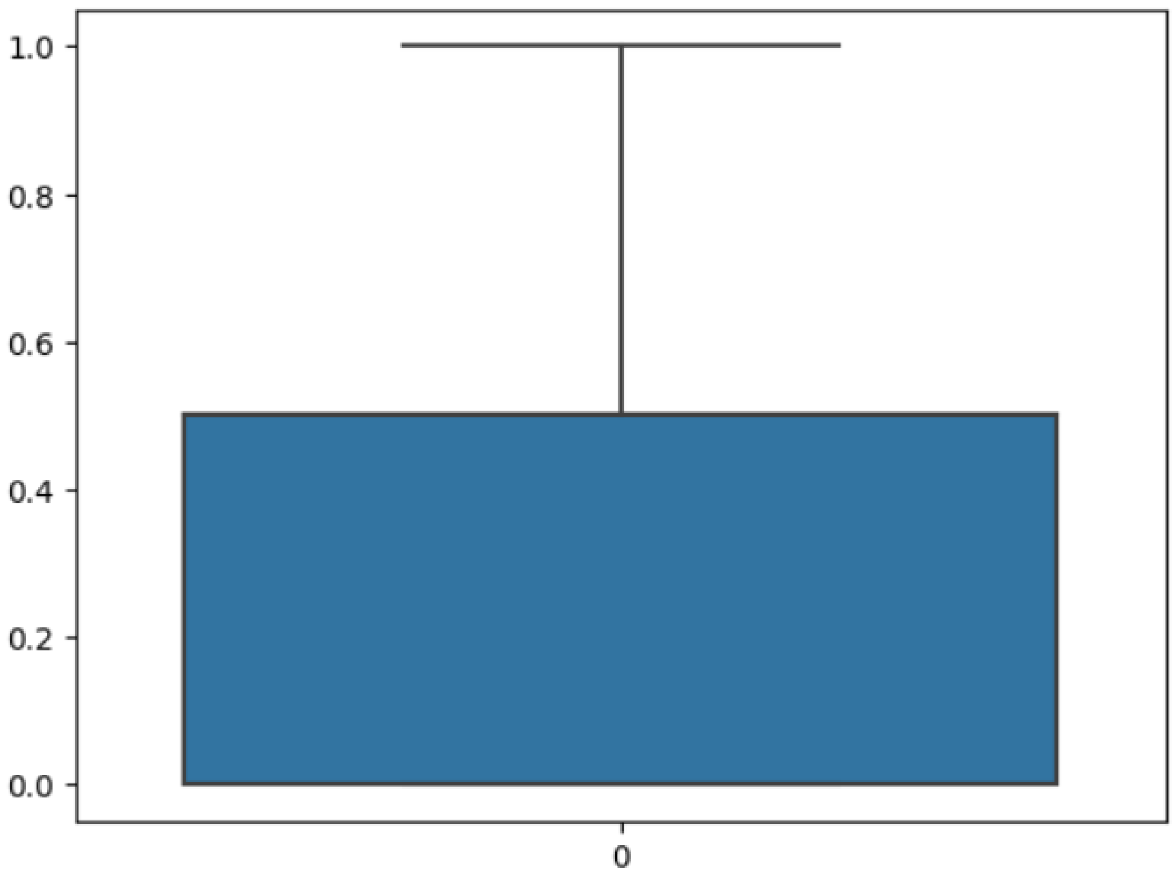
```
sns.boxplot(data['ST depression'])
```

```
<Axes: >
```



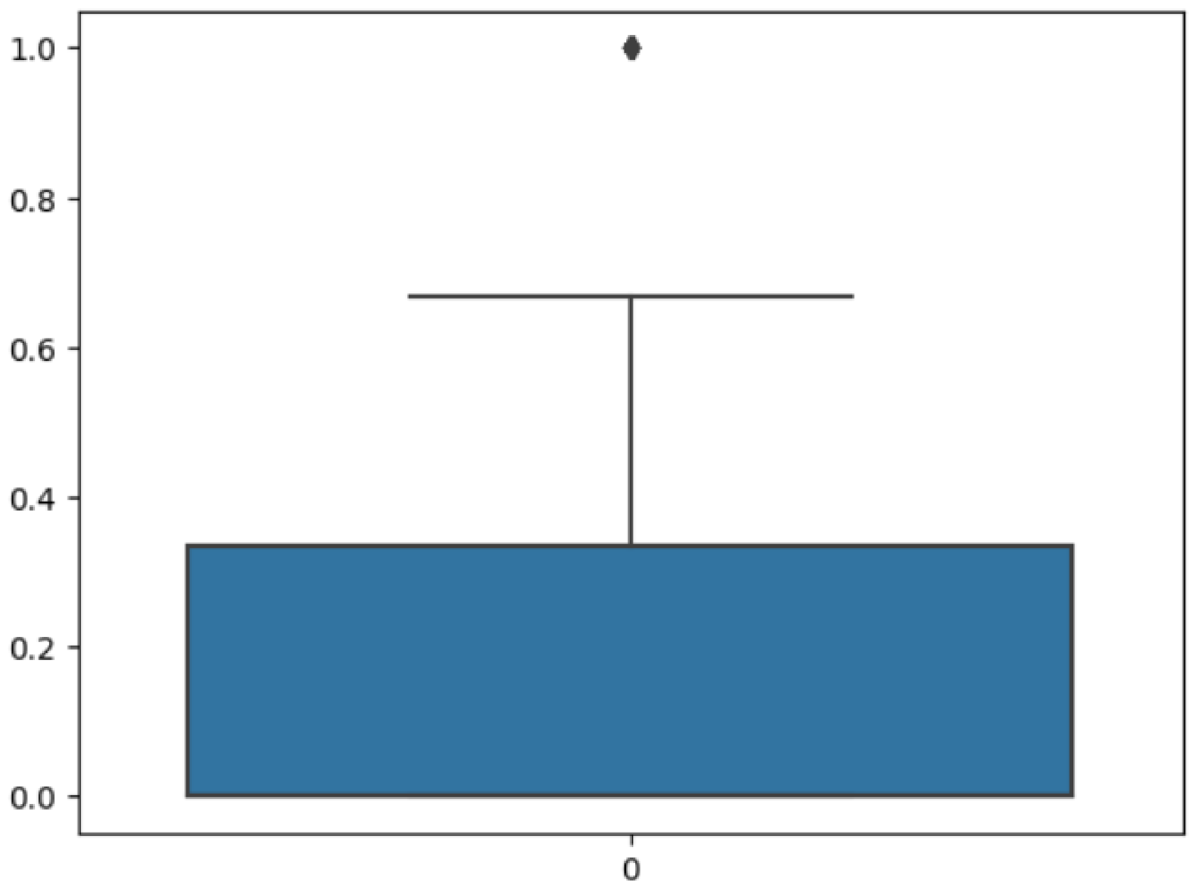
```
sns.boxplot(data['Slope of ST'])
```

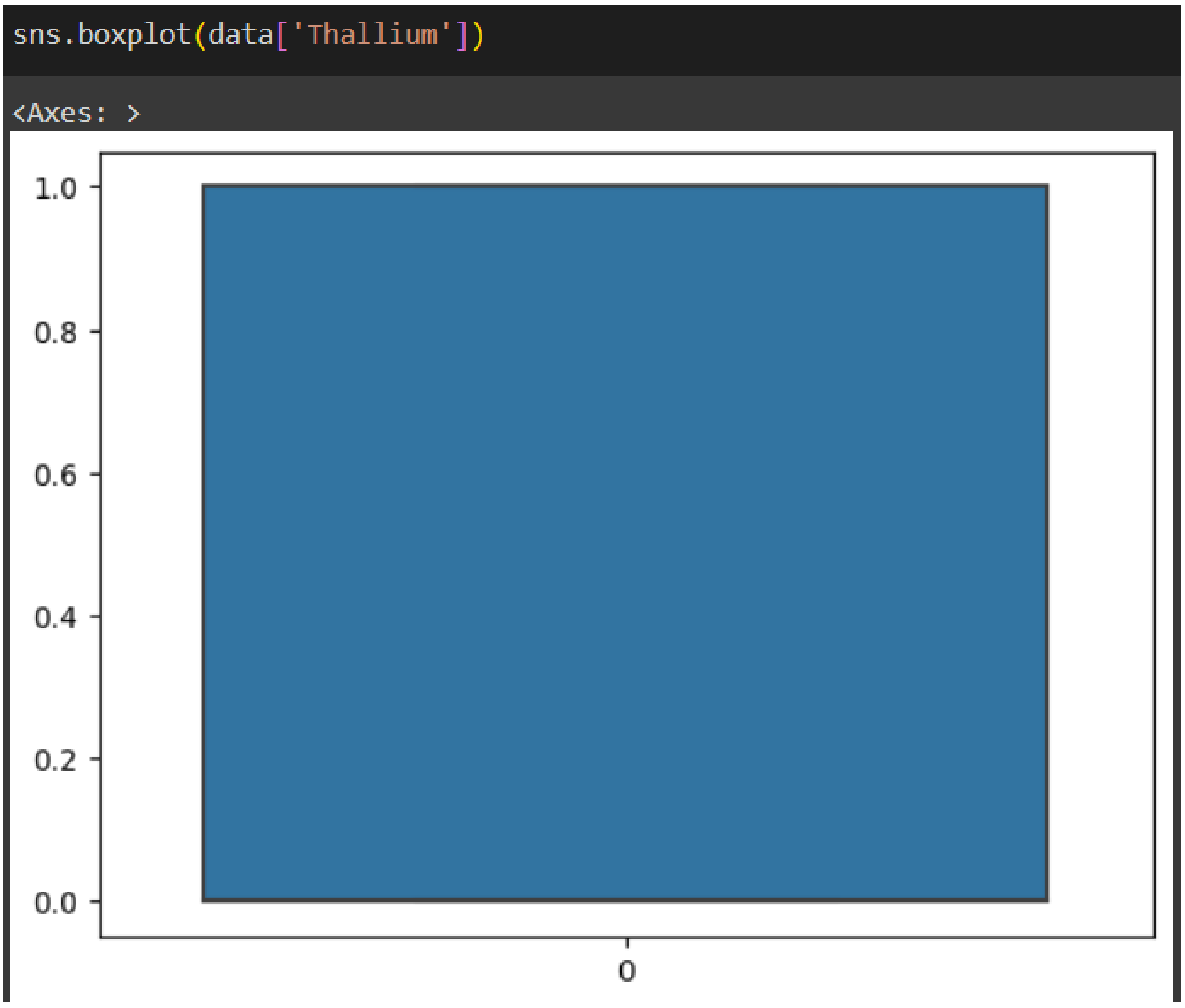
```
<Axes: >
```



```
sns.boxplot(data['Number of vessels fluro'])
```

```
<Axes: >
```





```
sns.boxplot(data['Thallium'])
```

```
<Axes: >
```

Models and hyperparameters used in the modeling step

1- **Train/test split:** done using train_test_split() function with hyperparameters of test_size = 0.2 and random_state = 44.

2- **Logistic Regression**: transforms its output using the logistic sigmoid function to return a probability value. logisticRegression() function from sklearn is used.

3- **SVM:** a binary classifier (a classifier used for those true/false, yes/no types of classification problems. Svc function is used from sklearn with hyperparameters of kernel='linear', C=1 and random_state=42.

4- **Decision Tree (ID3) models:** a decision tree is a structure that contains nodes (rectangular boxes) and edges(arrows) and is built from a dataset.ID3 stands for Iterative Dichotomiser 3 and is named such because the algorithm iteratively (repeatedly) dichotomizes(divides) features into two or more groups at each step. DecisionTreeClassifier() function was used and its hyperparameters are criterion='entropy', max_depth=5 and random_state=40

Conclusion

After preprocessing by replacing null values, replacing outliers and feature scaling, etc. then by using the best hyperparameters in the models A high accuracy was achieved in all models (87% in Logistic Regression, 81% in Decision Tree, 92.5% in SVM). From looking at this accuracy percentages SVM was the best model for this data.