Data Mining and Business Intelligence
CSE382

Heart Disease Prediction Algorithm
Classification Model

Phase 1

Team Members

Omar Baher Hussein	21P0315
Mohamed Khaled ElHelaly	21P0185
Mohamed Ehab Badr	21P0375
Saifeldin Mohamed Yousry	21P0362
Abdallah Ahmed Hamdy	21P0333
Ahmed Hossam ElDin	21P0271

Contents

Introduction

In recent years, cardiovascular diseases (CVDs) have become one of the leading causes of mortality worldwide, making early detection and prevention essential. The application of machine learning in healthcare offers significant potential to improve diagnostic accuracy and decision-making processes. This project focuses on predicting heart disease in patients using clinical data and understanding the role of outliers in the dataset.

Outliers are data points that deviate significantly from other observations and can impact the performance of machine learning models. Detecting and analyzing outliers is a critical preprocessing step in data science, as it helps ensure data quality and model reliability. In this project, we aim to:

1. Explore and understand a dataset containing clinical features relevant to heart disease.
2. Detect outliers in numeric features using two common techniques: **Interquartile Range (IQR)** and **Z-Score**.
3. Analyze the distribution of outliers across patients with and without heart disease to identify potential patterns.

Dataset Description

The dataset comprises 14 features, including clinical and diagnostic parameters, to predict the likelihood of heart disease in patients

Feature	Meaning
age	Patient's age
sex	Patient's sex (1 = male, 0 = female)
cp	Chest Pain Type • 0 = typical angina • 1 = atypical angina • 2 = non-anginal pain • 3 = asymptomatic
trestbps	Resting blood pressure (in mm Hg)
chol	Serum cholesterol (in mg/dl)
fbs	Fasting blood sugar > 120 mg/dl • 1 = true • 0 = false
restecg	Resting electrocardiographic results • 0 = normal • 1 = having ST-T wave abnormality • 2 = showing probable or definite left ventricular hypertrophy
thalach	Maximum heart rate achieved

exang	Exercise-induced angina • 1 = yes • 0 = no
oldpeak	ST depression induced by exercise relative to rest
slope	Slope of the peak exercise ST segment • 0 = upsloping • 1 = flat • 2 = downsloping
ca	Number of major vessels (0-3) colored by fluoroscopy
thal	Thalassemia • 1 = normal • 2 = fixed defect • 3 = reversible defect
target	Diagnosis of heart disease • 0 = no • 1 = yes

The dataset contains a mix of categorical and numeric features:

Numeric Features:

Continuous: These features take any real value.

- age
- trestbps
- chol
- thalach
- oldpeak

Categorical Features:

Ordinal: These features have a meaningful order but may not have consistent intervals.

- cp
- restecg
- slope
- ca
- thal

Nominal: These features have distinct categories with no intrinsic order.

- sex
- fbs
- exang
- target

Data Cleaning & Memory Optimization

Data Cleaning:

Missing values can negatively affect analysis and machine learning models. Dropping rows with missing values ensures the dataset is clean and ready for further processing.

```
heart_df = heart_df.dropna()
```

Memory Optimization:

Many machine learning workflows involve large datasets, and optimizing memory usage is crucial for performance.

Converting int64 (default for integer columns in pandas) to smaller types like int8 or int16 reduces memory usage significantly.

We reduced memory usage from 33KB to 6KB after applying optimization code:

```
heart_df = heart_df.astype({
    'age': 'int8',
    'sex': 'int8',
    'cp': 'int8',
    'trestbps': 'int16',
    'chol': 'int16',
    'fbs': 'int8',
    'restecg': 'int8',
    'thalach': 'int16',
    'exang': 'int8',
    'oldpeak': 'float32',
    'slope': 'int8',
    'ca': 'int8',
    'thal': 'int8',
    'target': 'int8'
})
```

Distribution Analysis:

Analyze the distribution of the target variable, which indicates the presence or absence of heart disease in the dataset. This helps us understand class distribution, assess dataset balance, and guide decisions for preprocessing and model evaluation.

```
print("\nDisease Presence Distribution:")
print(heart_df['target'].value_counts(normalize=True))
```

Disease Presence Distribution:

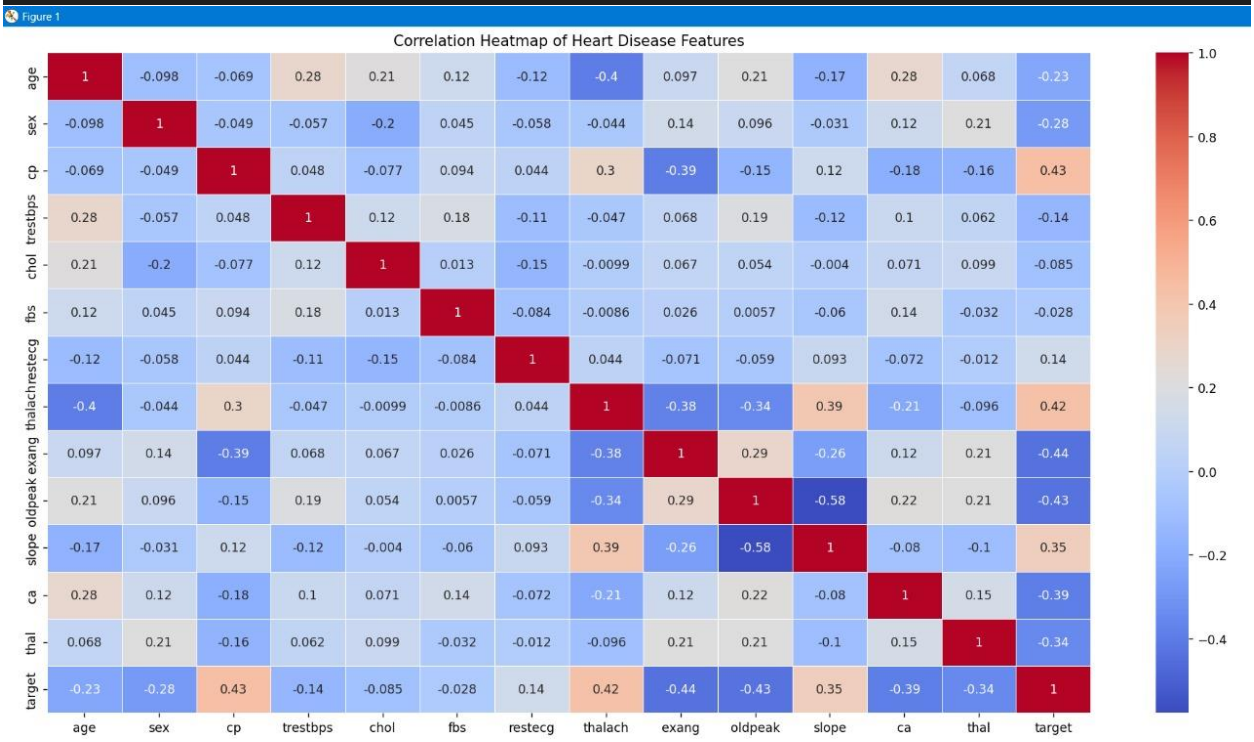
1 → 0.544554

0 → 0.455446

Correlation Between Features

Analyzing correlations between features is essential for designing efficient and interpretable models. It helps identify relationships, eliminate redundancy, and enhance feature selection, ultimately leading to better predictive performance and insights

```
plt.figure(figsize=(12, 10))
sns.heatmap(heart_df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap of Heart Disease Features')
plt.tight_layout()
plt.show()
```



The correlation heatmap visualizes the relationships between the various features in the heart disease dataset, including the target variable. Correlation values range between -1 and 1:

- +1 → Strong positive linear relationship
- 0 → No linear relationship.
- -1 → Strong negative linear relationship

Diagonal Values: All diagonal elements are 1 since a feature is perfectly correlated with itself.

Target Correlations: (target variable has notable correlations)

- Positive Correlation: cp (chest pain type) and thalach (maximum heart rate achieved) are positively correlated with the presence of heart disease.
- Negative Correlation: exang (exercise-induced angina), oldpeak (ST depression), and ca (number of major vessels) are negatively correlated with heart disease.

These correlations indicate that these features might be influential in predicting heart disease.

Relationships Between Features:

There is a strong positive correlation between oldpeak and exang, as well as between ca and oldpeak. These features might have related underlying mechanisms.
Features like sex and chol or trestbps and fbs show low or negligible correlation.

Detecting Outliers

Outliers are data points that significantly deviate from other observations in the dataset. They can influence statistical measures, distort machine learning models, and lead to inaccurate predictions. Hence, it is crucial to identify and handle them appropriately. In this project, outlier detection was performed using two common statistical techniques: **Interquartile Range (IQR)** and **Z-Score**. These methods were applied to numerical features in the dataset to identify data points that fall outside the expected range

```
numeric_features = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
methods = ['iqr', 'zscore']
for method in methods:
    for feature in numeric_features:
        FunctionSet.analyze_outlier_impact(feature, heart_df,output_dir=f'outliers/{method.upper()}',method=method)
    Outlier_Collector.outlier_collection(method=method)
```

Using IQR:

The get_outliers_iqr(df) function identifies and returns rows from a dataframe where the values in a specified column fall outside the acceptable range based on the Interquartile Range (IQR) method. It calculates the lower and upper bounds using the IQR and returns a filtered dataframe containing only the rows where the values fall outside these bounds in the specified column.

```
def get_outliers_iqr(df):
    Q1 = df[feature].quantile(0.25)
    Q3 = df[feature].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return df[(df[feature] < lower_bound) | (df[feature] > upper_bound)]
```

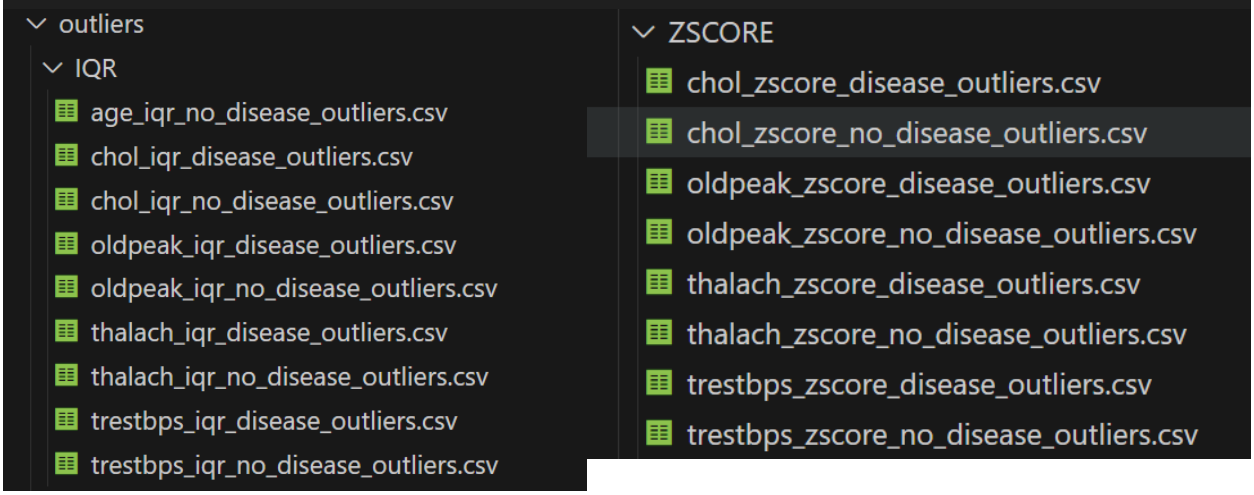
Using Z-Score:

The get_outliers_zscore(df, threshold=3) function identifies and returns rows from a dataframe where the values in a specified column deviate significantly from the mean, as determined by the Z-score method. It calculates the Z-score for each value and filters rows where the absolute Z-score exceeds the given threshold (default is 3). This helps detect outliers based on standard deviation.

```
def get_outliers_zscore(df, threshold=3):
    z_scores = np.abs((df[feature] - df[feature].mean()) / df[feature].std())
    return df[z_scores > threshold]
```

The save_outliers function saves identified outliers to CSV files. It takes two parameters: outliers, a DataFrame containing the outliers, and group_name, a string representing the group name (e.g., 'disease' or 'no_disease'). If the DataFrame is not empty, it constructs a file path using the output_dir, feature, method, and group_name variables, and saves the DataFrame to this file path as a CSV file without the index. It also prints a message indicating that the outliers have been saved. If the DataFrame is empty, it prints a message indicating that no outliers were found in the specified group.

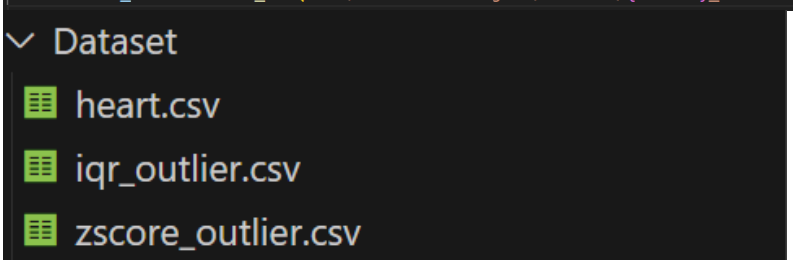
```python
def save_outliers(outliers, group_name):
    if not outliers.empty:
        file_path = os.path.join(output_dir, f'{feature}_{method}_{group_name}_outliers.csv')
        outliers.to_csv(file_path, index=False)
        print(f"{group_name.capitalize()} group outliers saved to '{file_path}'")
    else:
        print(f"No outliers found in {group_name} group.")
```



Outlier Collection

The outlier_collection(method) function aggregates outlier data files from a specified directory based on the selected method (IQR or Z-Score). It first reads an initial CSV file corresponding to the method and then iterates through the directory to load and concatenate additional CSV files, skipping the first iteration and including only files with a .csv extension. The combined DataFrame is saved as a single CSV file in the specified output directory, allowing all outlier data to be collected into 1 file for each method for further use cases.

```python
def outlier_collection(method):
    global first_file
    directory = os.scandir(f'outliers/{method.upper()}')
    if method == 'iqr':
        first_file = pd.read_csv(f'outliers/{method.upper()}/age_iqr_no_disease_outliers.csv')
    elif method == 'zscore':
        first_file = pd.read_csv(f'outliers/{method.upper()}/chol_zscore_disease_outliers.csv')
    Outlier_dataframe = first_file
    i = 0

    for file in directory:
        if i == 0:
            i += 1
            continue
        if file.name.endswith('.csv'):
            print(file.name)
            df = pd.read_csv(f'outliers/{method.upper()}/' + file.name)
            Outlier_dataframe = pd.concat([Outlier_dataframe, df], ignore_index=True)

    Outlier_dataframe.to_csv(f'../DM-and-Bi-Project/Dataset/{method}_outlier.csv', index=False)
```



Cleaning the Dataset:

The Dataset now removes any row that matches the data in the df_outlier dataframe. The updated dataframe, which excludes the outliers, is then saved to a new CSV file called heart_no_outliers.csv. This ensures that the final dataset does not contain the specified outlier data, making it suitable for further analysis.

```python
data = pd.read_csv('Dataset/heart.csv')
df_outlier = pd.read_csv('iqr_outliers.csv')

for i in data.index.copy():
    for j in df_outlier.index:
        if data.loc[i].equals(df_outlier.loc[j]):
            data = data.drop(i)  # Drop the matching row
            break

data.to_csv('Dataset/heart_no_outliers.csv', index=False)
```

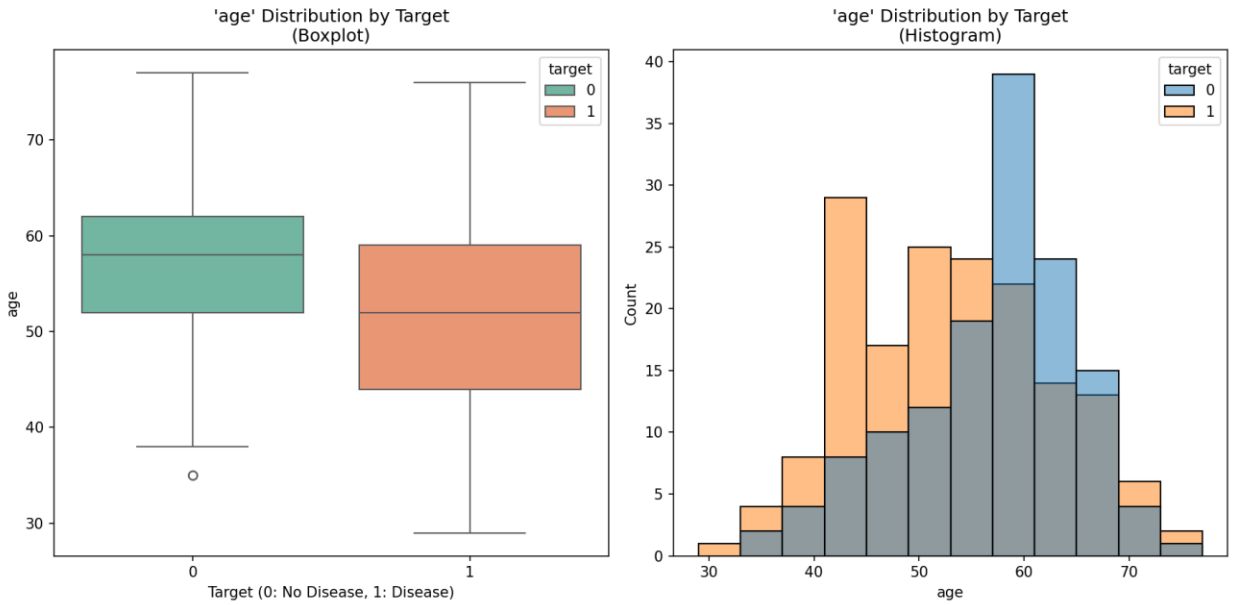
Outlier Visualization

Analyzing and comparing how IQR and Z-Score differ in detecting outliers involves visualizing the distribution of data and identifying data points that fall outside the thresholds set by each method.

```
# Create visualization
plt.figure(figsize=(12, 6))

# Create subplot for boxplot
plt.subplot(1, 2, 1)
sns.boxplot(x='target', y=feature, data=df, hue='target', palette='Set2', dodge=False)
plt.title(f"'{feature}' Distribution by Target\n(Boxplot)")
plt.xlabel('Target (0: No Disease, 1: Disease)')
plt.ylabel(feature)

# Create subplot for distribution plot
plt.subplot(1, 2, 2)
sns.histplot(data=df, x=feature, hue='target', multiple="layer", alpha=0.5)
if method.lower() == 'zscore':
    mean = df[feature].mean()
    std = df[feature].std()
    plt.axvline(mean - 3*std, color='r', linestyle='--', alpha=0.5, label='±3 SD')
    plt.axvline(mean + 3*std, color='r', linestyle='--', alpha=0.5)
plt.title(f"'{feature}' Distribution by Target\n(Histogram)")
plt.xlabel(feature)
plt.ylabel('Count')

plt.tight_layout()
plt.show()
```

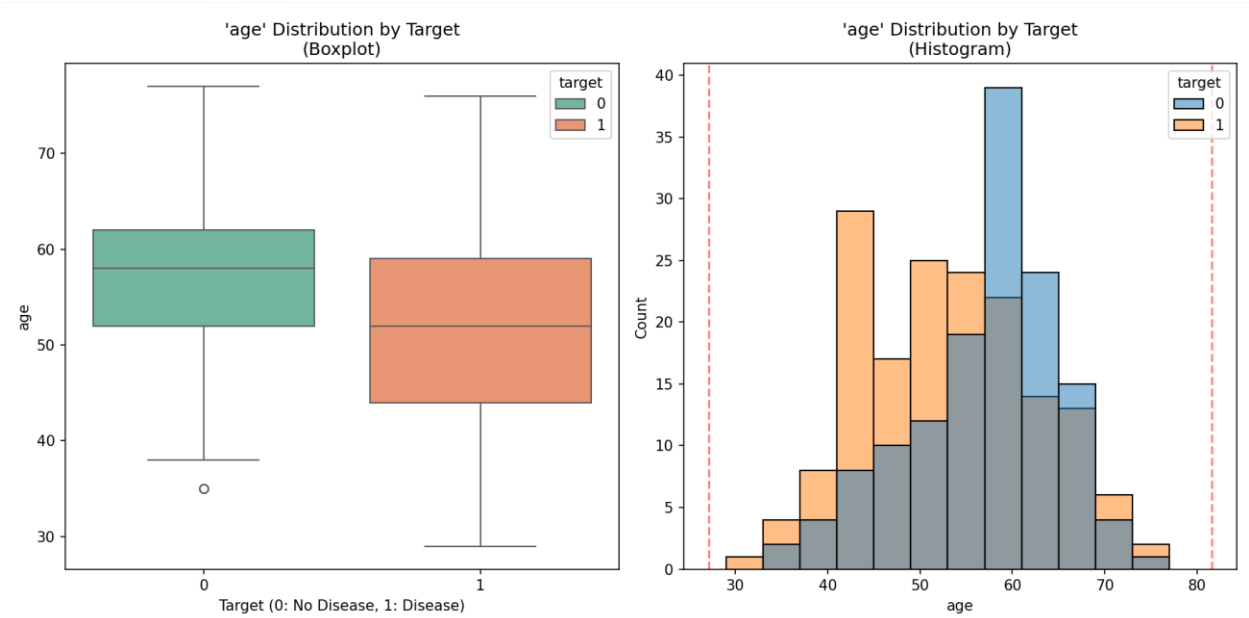
Age Feature Outlier Detection using IQR:

- Disease Group: 0 outliers out of 165 records (0.00%)
- No Disease Group: 2 outliers out of 138 records (1.45%)



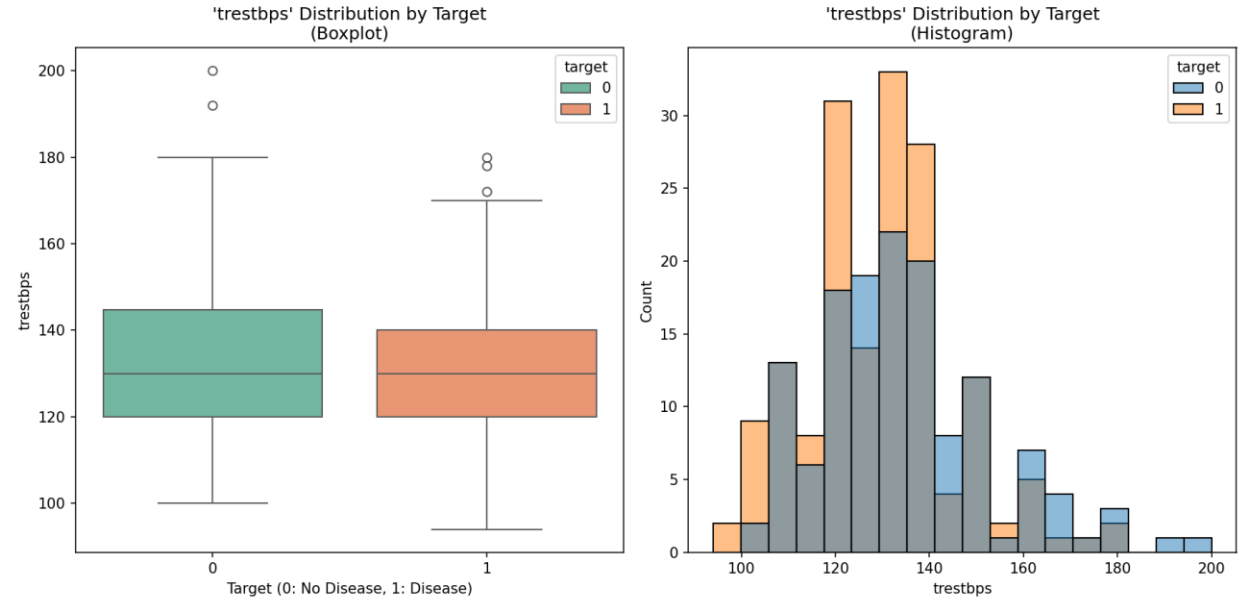
Age Feature Outlier Detection using Z-Score:

- Disease Group: 0 outliers out of 165 records (0.00%)
- No Disease Group: 0 outliers out of 138 records (0.00%)



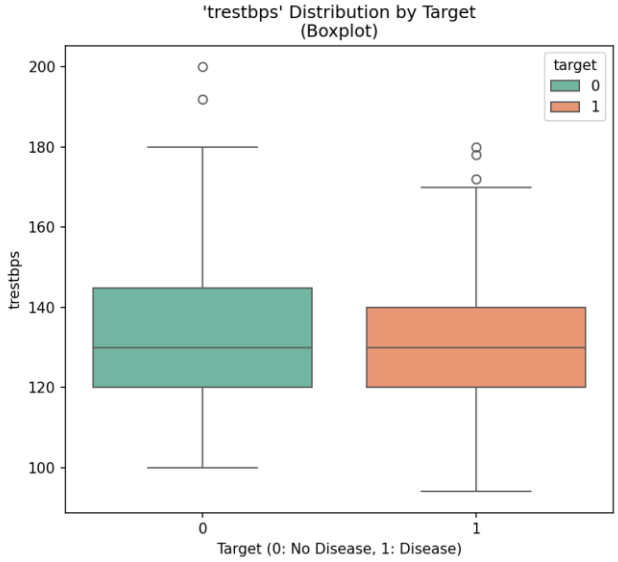
Trestbps Feature Outlier Detection using IQR:

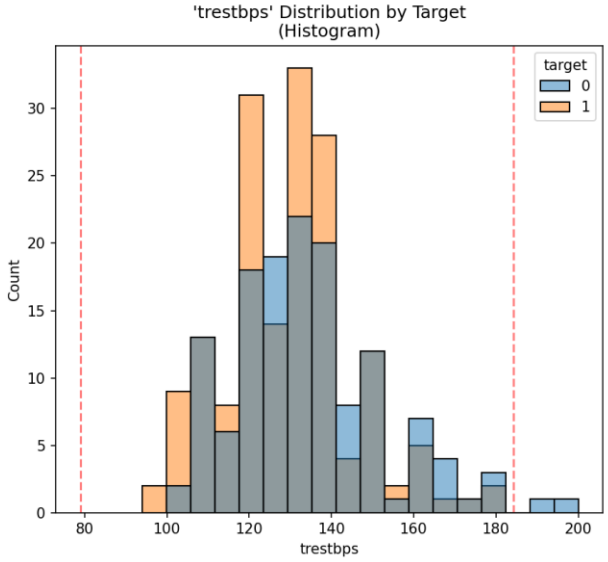
- Disease Group: 3 outliers out of 165 records (1.82%)
- No Disease Group: 2 outliers out of 138 records (1.45%)



Trestbps Feature Outlier Detection using Z-Score:

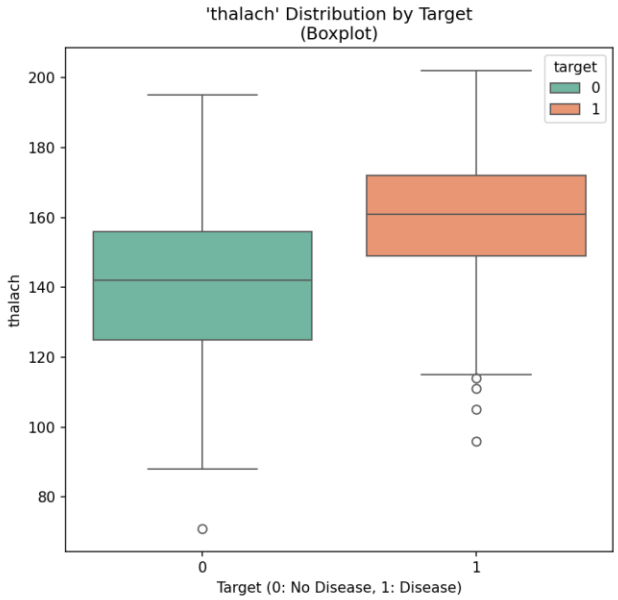
- Disease Group: 2 outliers out of 165 records (1.21%)
- No Disease Group: 2 outliers out of 138 records (1.45%)

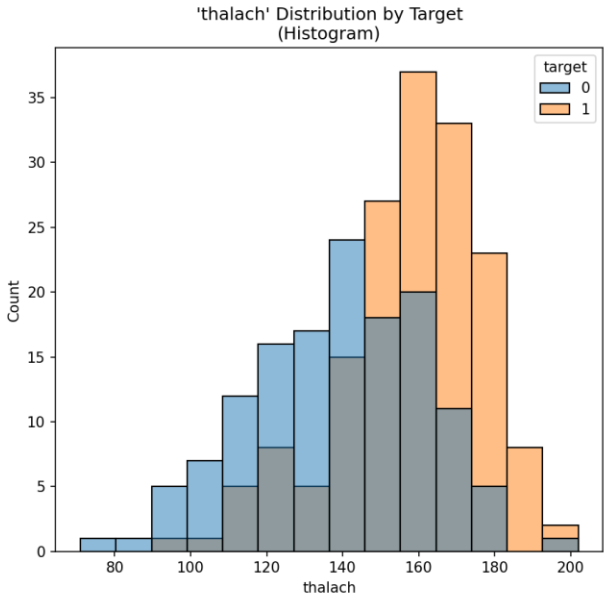




Thalac Feature Outlier Detection using IQR:

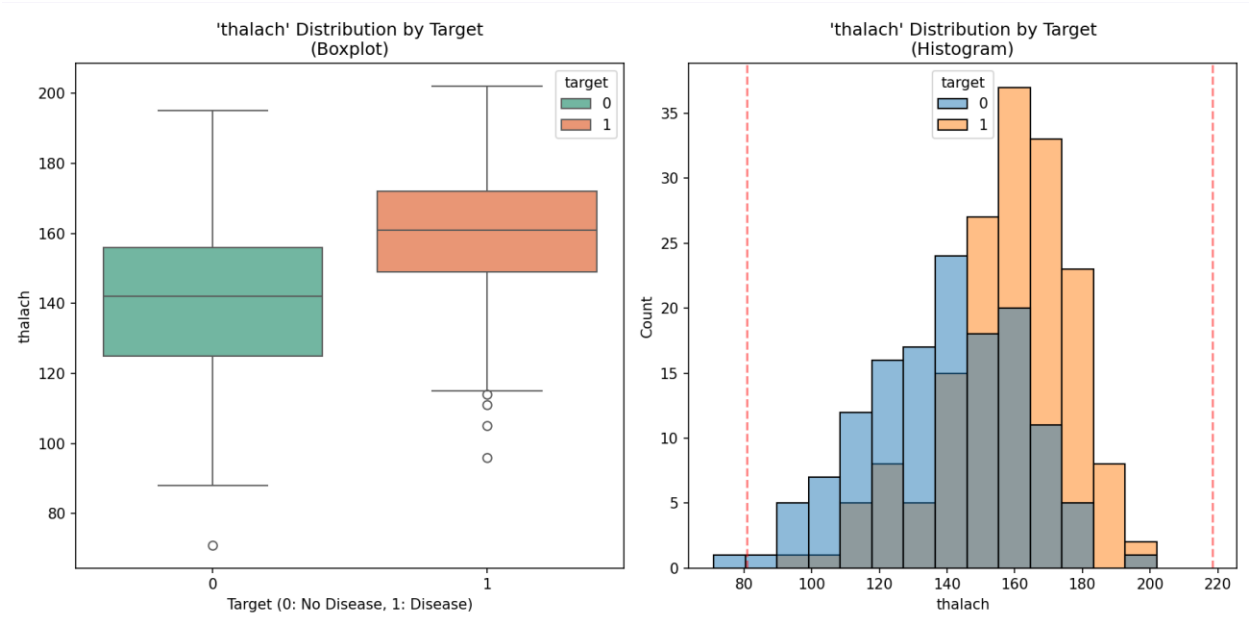
- Disease Group: 4 outliers out of 165 records (2.42%)
- No Disease Group: 1 outlier out of 138 records (0.72%)





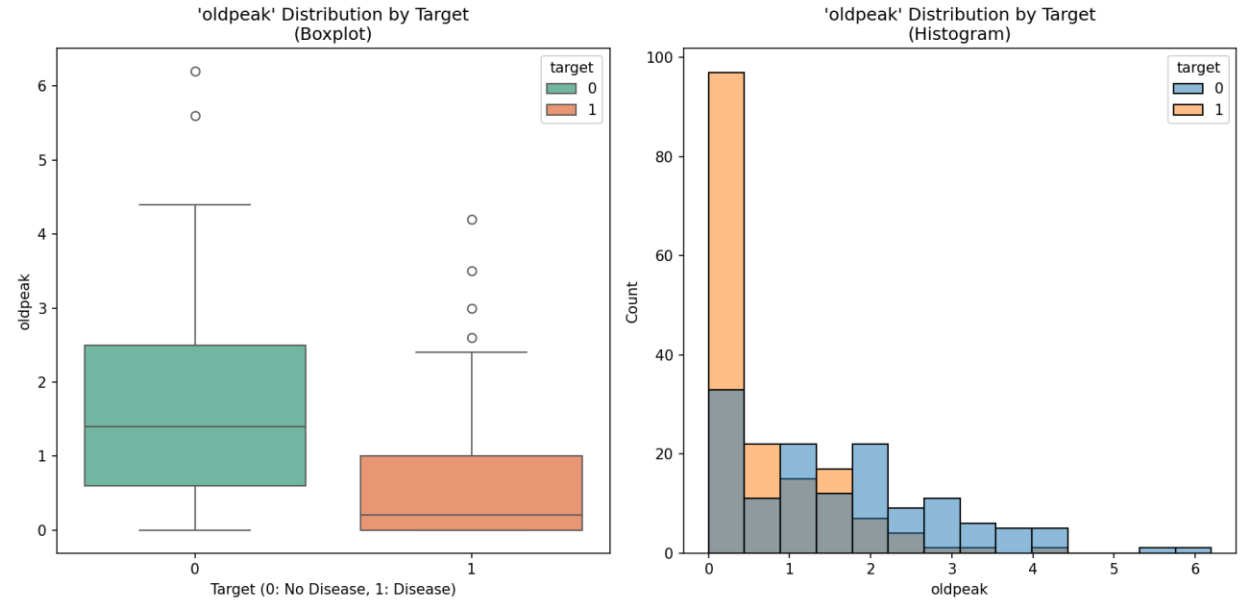
Thalac Feature Outlier Detection using Z-Score:

- Disease Group: 1 outlier out of 165 records (0.61%)
- No Disease Group: 1 outlier out of 138 records (0.72%)



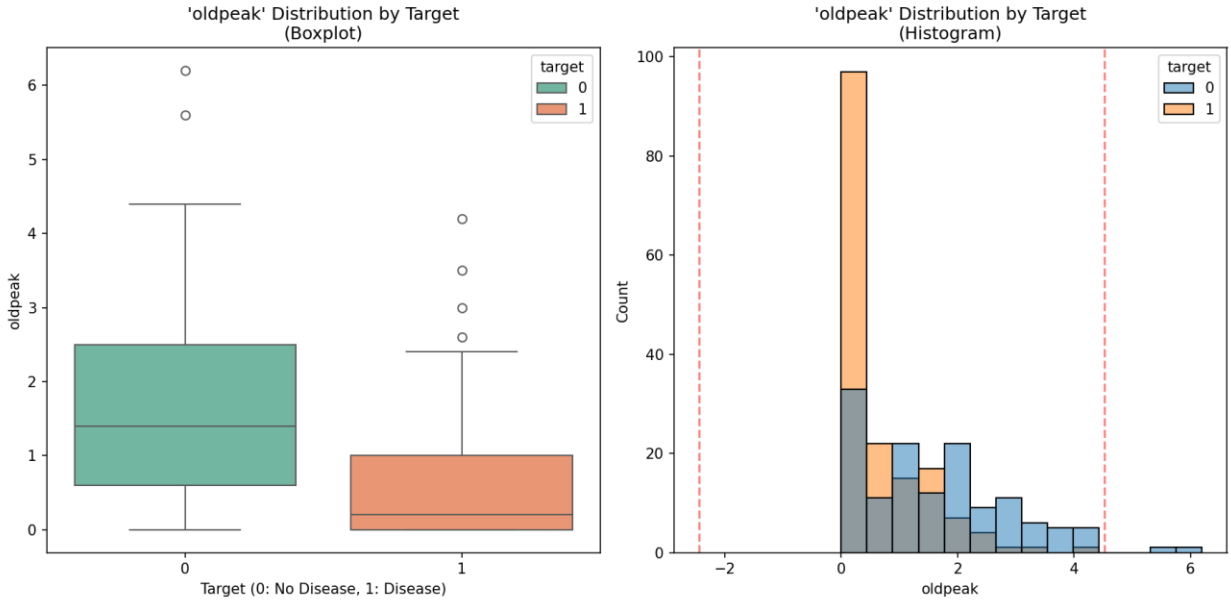
OldPeak Feature Outlier Detection using IQR:

- Disease Group: 4 outliers out of 165 records (2.42%)
- No Disease Group: 2 outliers out of 138 records (1.45%)



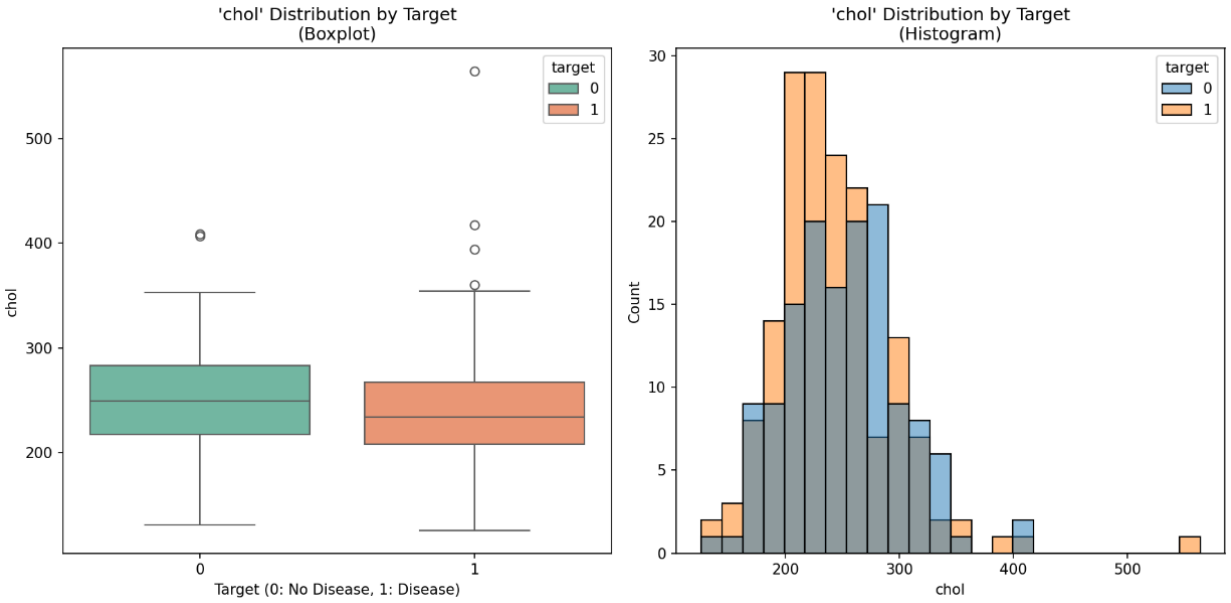
OldPeak Feature Outlier Detection using Z-Score:

- Disease Group: 3 outliers out of 165 records (1.82%)
- No Disease Group: 2 outliers out of 138 records (1.45%)



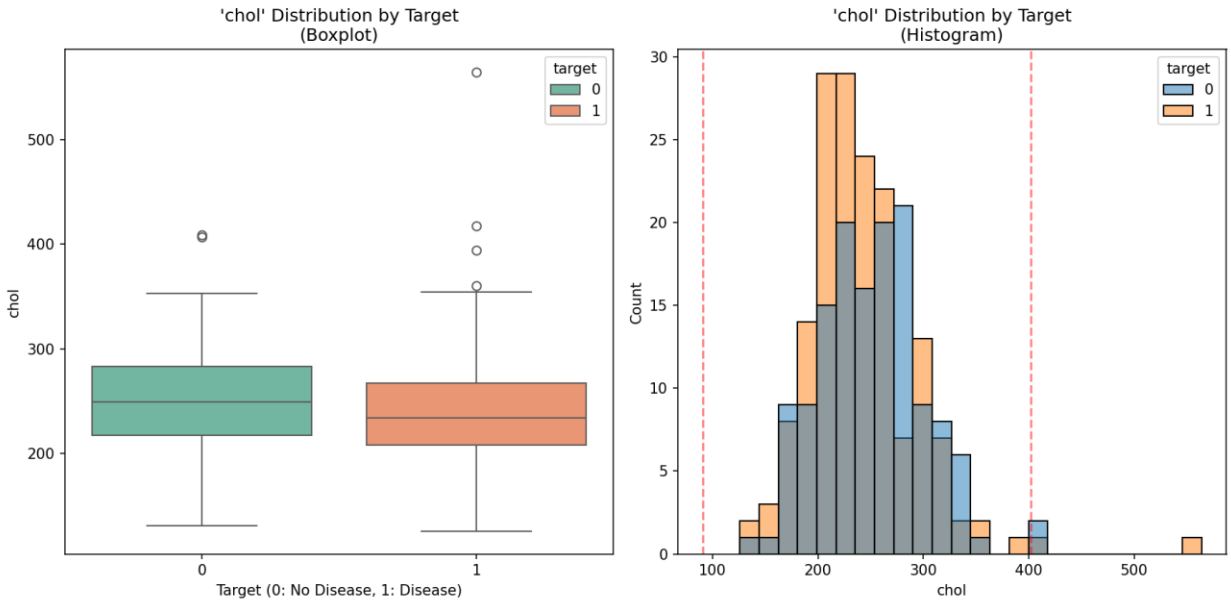
Cholesterol Feature Outlier Detection using IQR:

- Disease Group: 4 outliers identified out of 165 records (2.42%).
- No Disease Group: 2 outliers identified out of 138 records (1.45%).



Cholesterol Feature Outlier Detection using Z-Score:

- Disease Group: 2 outliers identified out of 165 records (1.21%).
- No Disease Group: 2 outliers identified out of 138 records (1.45%).



Conclusion

For the all the features, the **IQR method** was more effective at detecting outliers in the disease group, suggesting it may be better suited for datasets where more conservative outlier detection is needed. The Z-Score method performed similarly to the IQR method in the no disease group but was less effective for the disease group. Therefore, the IQR method was preferred over the Z-Score method.