Altrium Bootcamp - Assignment 1

REPORT BY: ANUTTARA RAJASINGHE

Task 1 : Data Exploration

For this task I'm encouraged to explore the dataset and perform various operations such as cleaning, feature selection, and feature engineering to prepare the data for analysis.

1. Loading and Exploring the Dataset

- The code begins by importing the necessary libraries, pandas and numpy.
- The processed dataset "processed.cleveland.csv" is loaded into a pandas DataFrame using the read_csv() function.
- The first few rows of the dataset are displayed using the head() function to get an initial understanding of the data

2. Assigning Column Names

- A list of column names is created to provide meaningful labels to the dataset.
- The column names are assigned to the DataFrame using the columns attribute.
- The DataFrame is displayed again using the head() function to show the dataset with updated column names.

3. Generating an Exploratory Data Analysis (EDA) Report

- The Sweetviz library is imported as sv.
- The analyze() function from Sweetviz is used to generate an EDA report on the dataset.
- The generated report is saved as an HTML file using the show_html() function.
- This report is saved in the git repository as report.html.

4. Data Cleaning: Replacing "?" with NaN and Dropping Null Values

- The replace() function is used to replace "?" values in the dataset with NaN.
- The number of records in the dataset before dropping null values is displayed as 303.
- The dropna() function is used to drop rows containing null values from the dataset.
- The number of records in the dataset after dropping null values is displayed as 297.

5. Data Cleaning: Dropping Duplicate Values

- The number of records in the dataset before dropping duplicate values is displayed as 297
- The drop_duplicates() function is used to drop duplicate rows from the dataset.
- The number of records in the dataset after dropping duplicate values is displayed as 297 indicating that there weren't any duplicate values.

6. Summary Statistics of the Cleaned Dataset

- The cleaned dataset is displayed using the head() function to show the updated data.
- Descriptive statistics of the cleaned dataset are displayed using the describe() function.
- Information about the cleaned dataset, including column names, non-null counts, and data types, is displayed using the info() function.

7. Plotting a Correlation Matrix for Analyzing Features

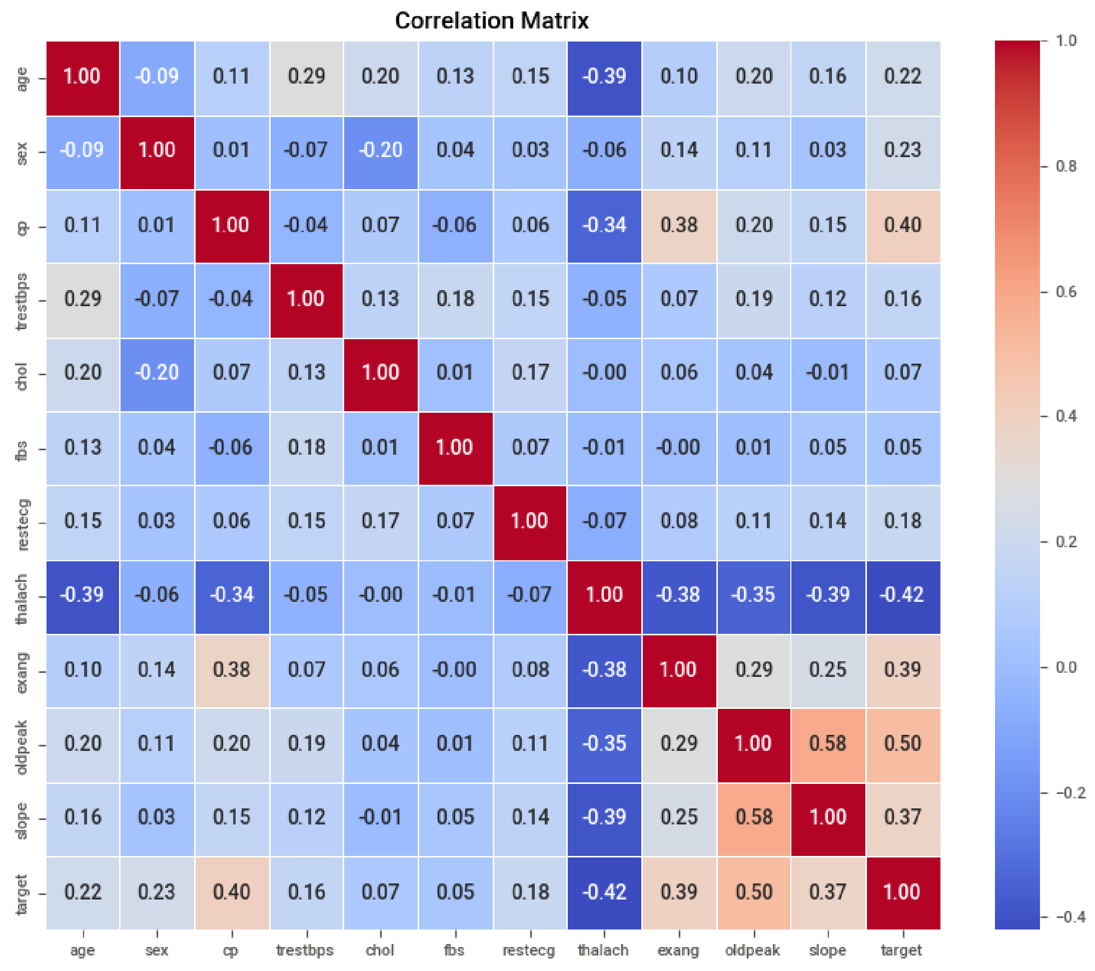
- The matplotlib and seaborn libraries are imported for data visualization.
- The correlation matrix of the cleaned dataset is computed using the corr() function.
- The correlation matrix is plotted as a heatmap using the heatmap() function from seaborn.

8. Feature Selection based on Correlation

- A correlation threshold of 0 is set.
- The correlation between each feature and the target variable is computed.
- Features with an absolute correlation greater than the threshold are selected. Only 12 out of the 14 features are taken into consideration.
- A new DataFrame, new_clev_DF, is created containing only the selected features.
- The selected features and their corresponding summary are displayed using the head() and columns.tolist() functions.

Task 2 : Data Analysis

1. Plotting the correlation matrix

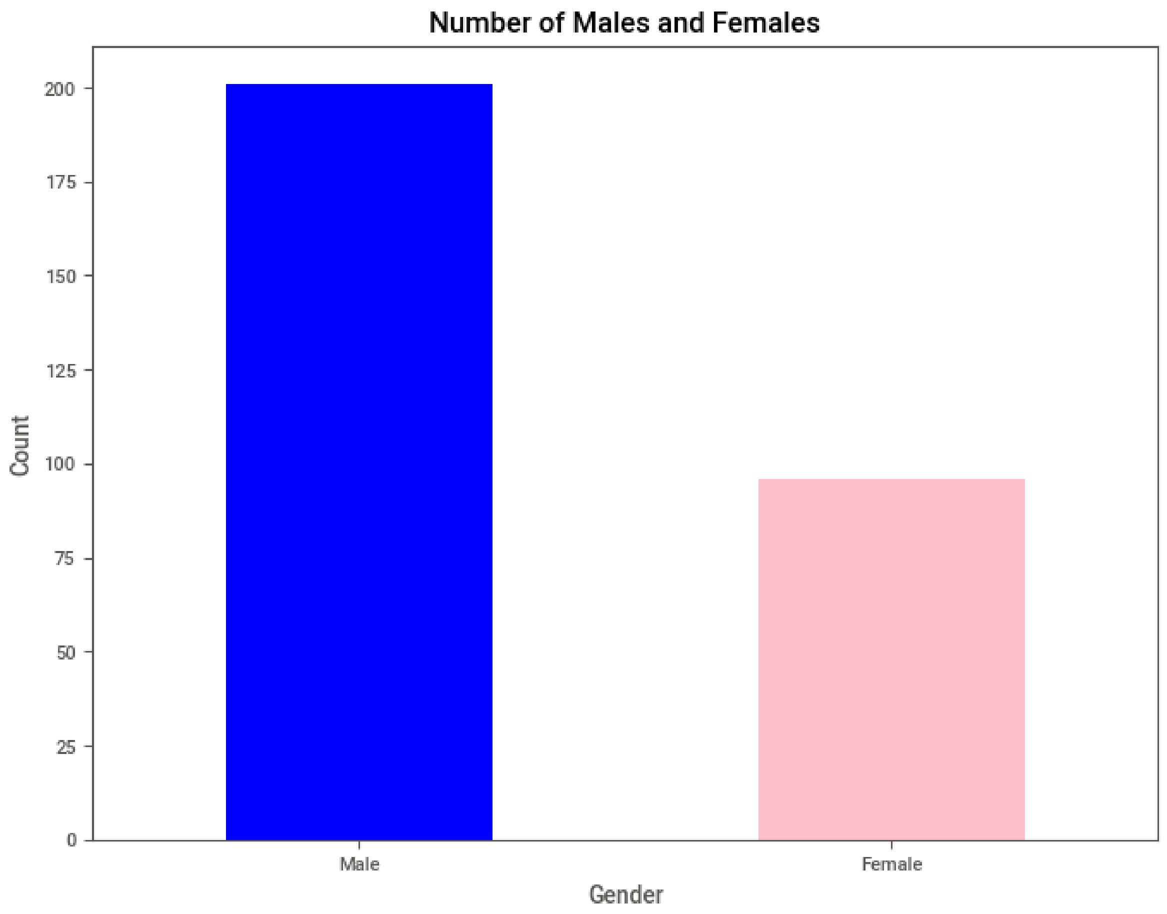


	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	target
age	1.00	-0.09	0.11	0.29	0.20	0.13	0.15	-0.39	0.10	0.20	0.16	0.22
sex	-0.09	1.00	0.01	-0.07	-0.20	0.04	0.03	-0.06	0.14	0.11	0.03	0.23
cp	0.11	0.01	1.00	-0.04	0.07	-0.06	0.06	-0.34	0.38	0.20	0.15	0.40
trestbps	0.29	-0.07	-0.04	1.00	0.13	0.18	0.15	-0.05	0.07	0.19	0.12	0.16
chol	0.20	-0.20	0.07	0.13	1.00	0.01	0.17	-0.00	0.06	0.04	-0.01	0.07
fbs	0.13	0.04	-0.06	0.18	0.01	1.00	0.07	-0.01	-0.00	0.01	0.05	0.05
restecg	0.15	0.03	0.06	0.15	0.17	0.07	1.00	-0.07	0.08	0.11	0.14	0.18
thalach	-0.39	-0.06	-0.34	-0.05	-0.00	-0.01	-0.07	1.00	-0.38	-0.35	-0.39	-0.42
exang	0.10	0.14	0.38	0.07	0.06	-0.00	0.08	-0.38	1.00	0.29	0.25	0.39
oldpeak	0.20	0.11	0.20	0.19	0.04	0.01	0.11	-0.35	0.29	1.00	0.58	0.50
slope	0.16	0.03	0.15	0.12	-0.01	0.05	0.14	-0.39	0.25	0.58	1.00	0.37
target	0.22	0.23	0.40	0.16	0.07	0.05	0.18	-0.42	0.39	0.50	0.37	1.00

This calculates the correlation matrix for the dataset and visualizes it using a heatmap. It starts by computing the correlation matrix using the corr() function on the new_clev_DF dataframe. Then, it sets the figure size for the plot and creates a heatmap using sns.heatmap(). The heatmap is annotated with correlation9 values, uses the 'coolwarm' colormap, and displays correlation values with two decimal places. Finally, the plot is titled 'Correlation Matrix' and displayed using plt.show().

This maps out the correlation between each variable and that shows how relevant these variables are to one another.

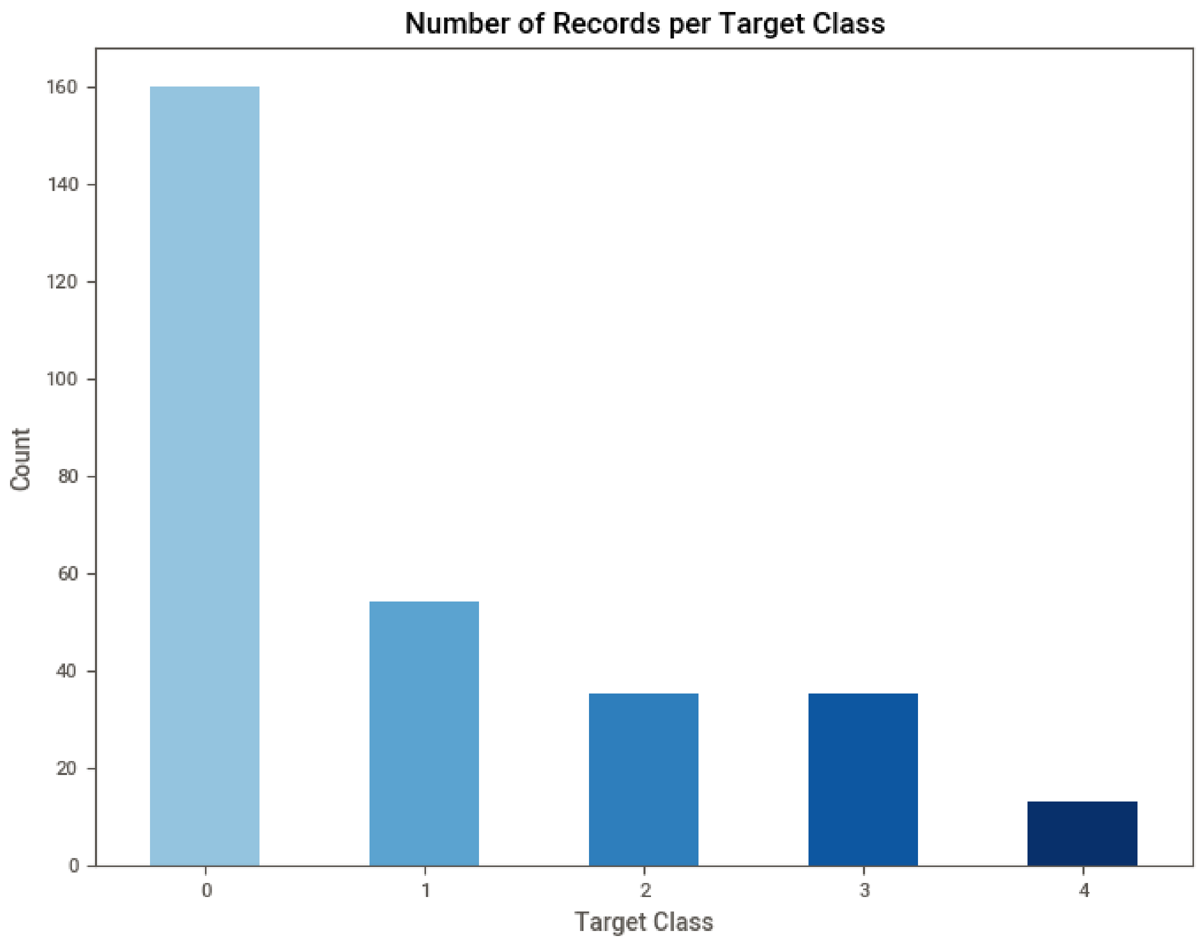
2. Comparing the records taken by males and females



This chart compares the number of records for males and females in the dataset. It begins by counting the occurrences of each gender using the value_counts() function on the 'sex' column of the new_clev_DF dataframe. Then, it creates a bar plot to visualize the counts, with blue for males and pink for females. The plot is labeled with 'Gender' on the x-axis and 'Count' on the y-axis, and titled 'Number of Males and Females'. The x-tick labels are set to 'Male' and 'Female', and the plot is displayed using plt.show().

As we can see, the number of male records in this dataset is much higher than the number of female records in this dataset, this may lead to bias when developing machine learning models. A solution can be to either oversample the female records or undersample the male records.

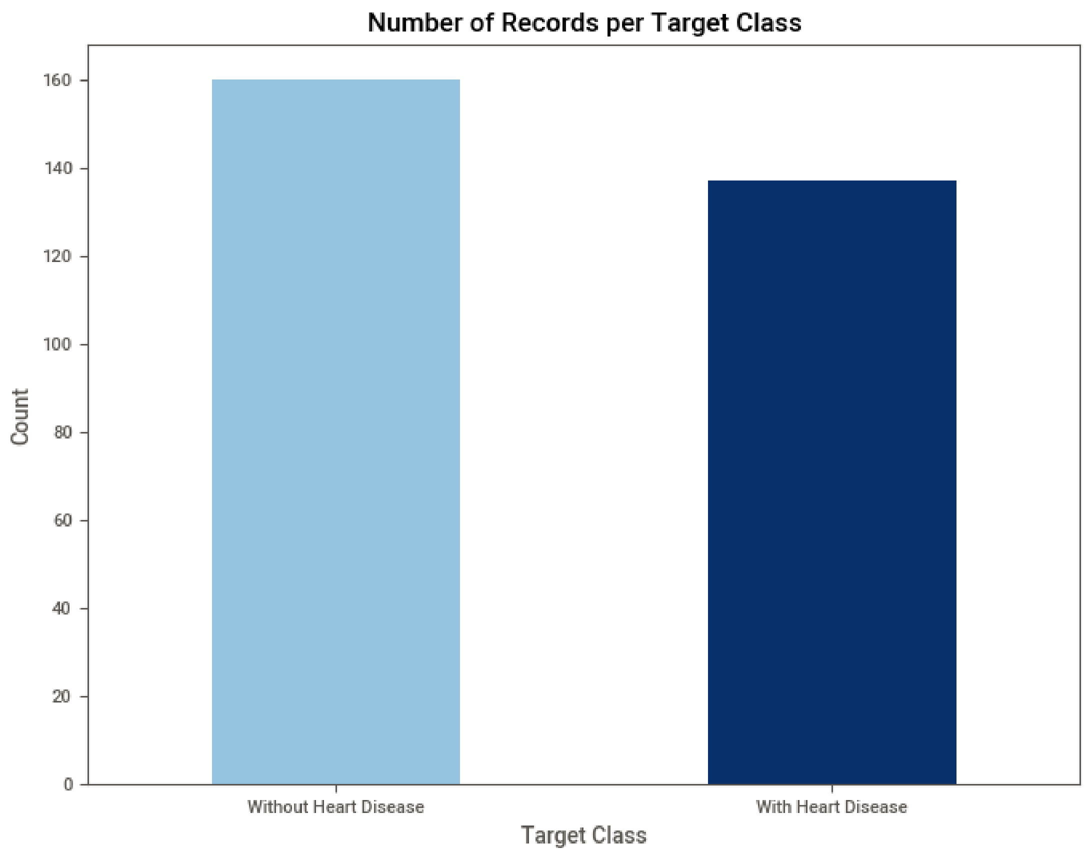
3. Counting the number of records for classes of target variable



In this, the number of records for each class of the target variable is counted. It starts by counting the occurrences of each target class using the value_counts() function on the 'target' column of the new_clev_DF dataframe. The colors for the bar plot are generated using a colormap ('Blues') and a range of values. Then, a bar plot is created to visualize the counts, with different shades of blue representing each target class. The plot is labeled with 'Target Class' on the x-axis and 'Count' on the y-axis, and titled 'Number of Records per Target Class'. The x-tick labels are not rotated, and the plot is displayed using plt.show().

As per the documentation: "It is integer valued from 0 (no presence) to 4. Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0)." According to this, it shows that the number of records with no heart disease is much higher than the records with heart disease present and the last being the highest stage of heart disease, an uneven distribution.

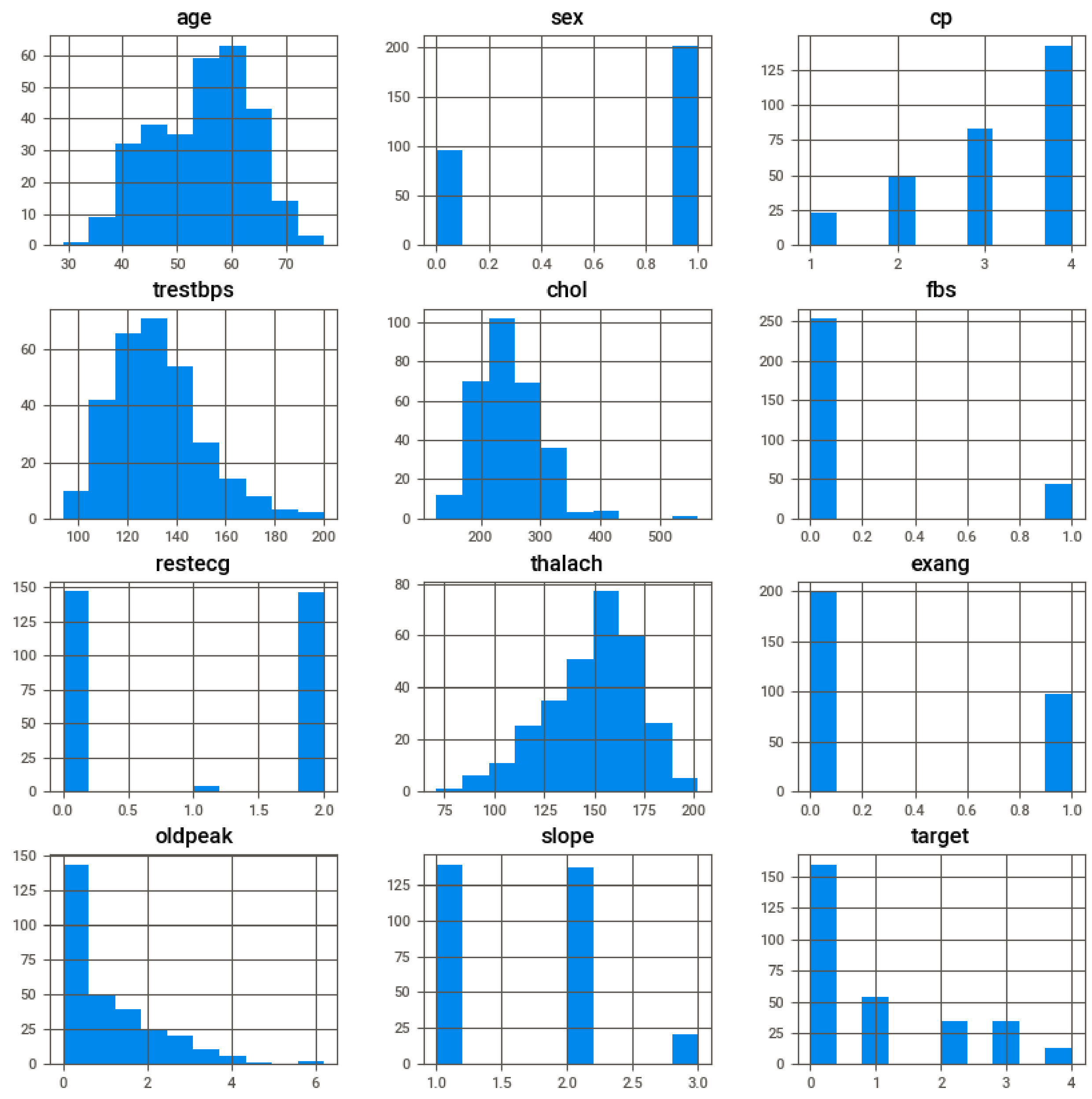
4. Summarized Target Plot



This creates a summarized target plot, mapping the target values to descriptive labels. It starts by creating a new dataframe plot_df with only the 'target' column from new_clev_DF. Then, the 'target' values are replaced with descriptive labels using the replace() function. The counts for each target class are computed using value_counts(), and the colors for the bar plot are generated using a colormap ('Blues') and a range of values. Finally, a bar plot is created to visualize the counts, with different shades of blue representing each target class. The plot is labeled with 'Target Class' on the x-axis and 'Count' on the y-axis, and titled 'Number of Records per Target Class'. The x-tick labels are not rotated, and the plot is displayed using plt.show().

This shows the comparison of the count of data for people with and without heart disease. This shows that the total number of people without heart disease is higher than the total number of people with heart disease. This edges towards bias in the dataset towards the “With Heart Disease” category and can be solved by undersampling the “Without Heart Disease” category or oversampling the “With heart Disease” category to provide an even, unbiased dataset for training a certain model.

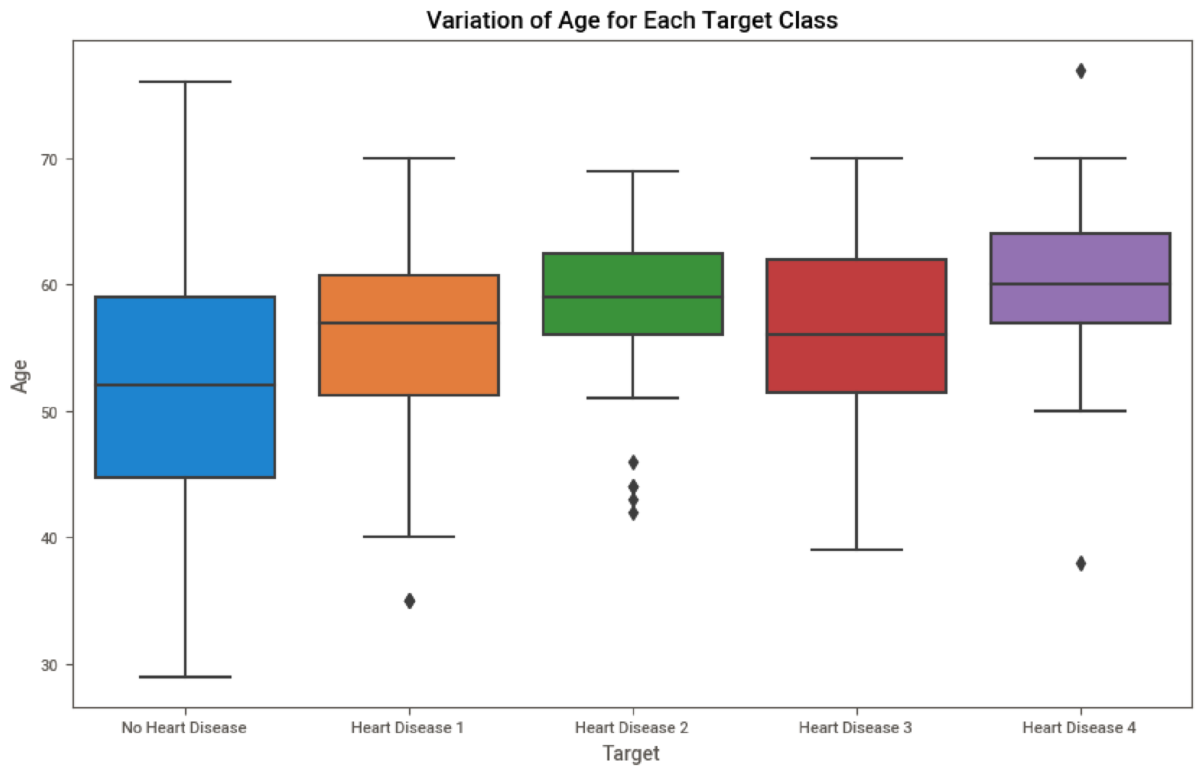
5. Histograms of Cleveland data



This creates histograms for each feature in the dataset. It uses the hist() function on the new_clev_DF dataframe with a specified figure size to create a grid of histograms. The resulting plot displays histograms for each feature, arranged in a 3x4 grid. Each subplot is labeled with the feature name as the title.

This shows the distribution of datasets in all features, represented in the form of histograms. This is for all 12 features.

6. Age variation for target values

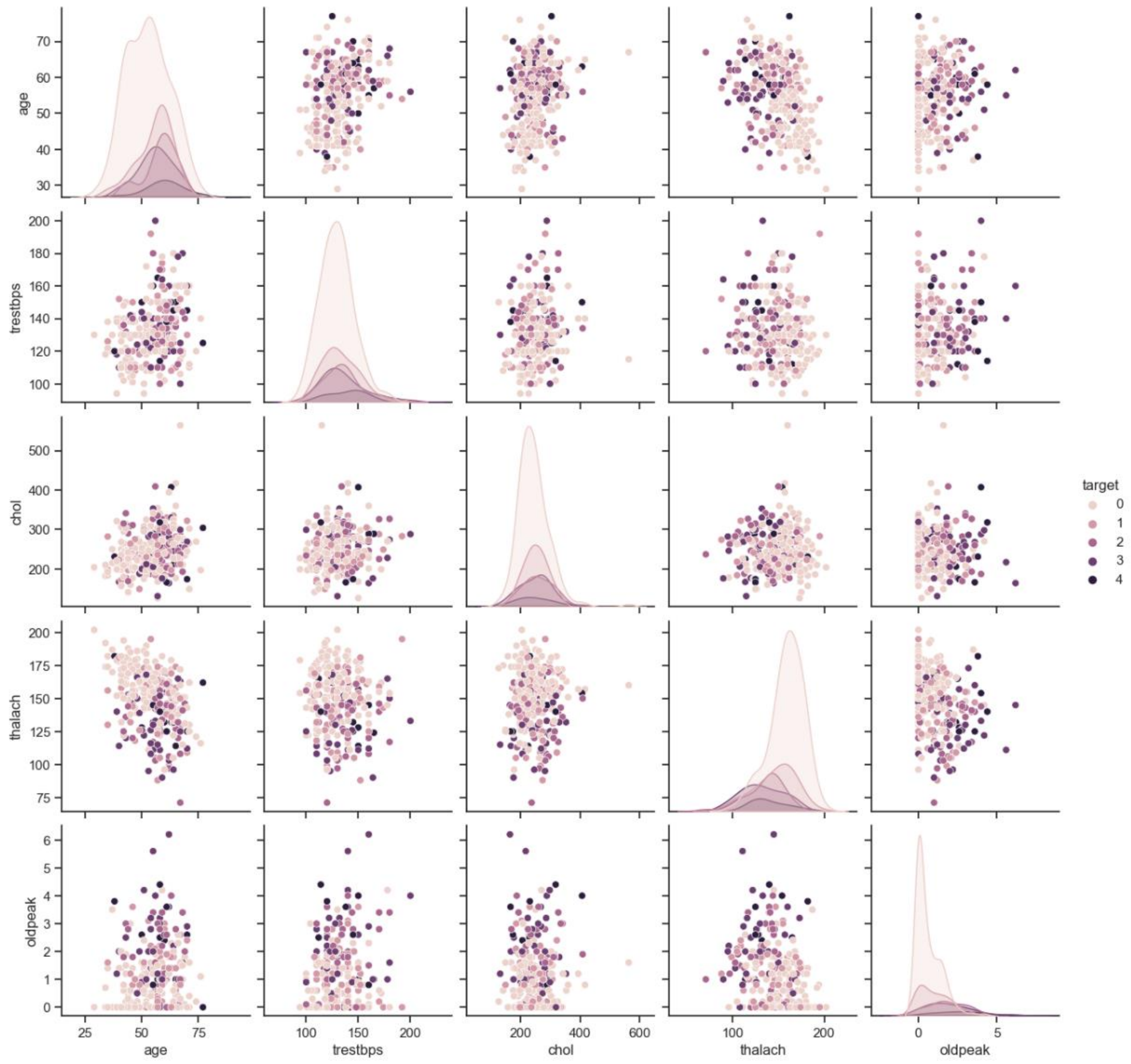


A boxplot is created using the boxplot() function to visualize the variation of age for each target class. The x-axis represents the target classes, and the y-axis represents the age values. The plot provides insights into the distribution of age within different target classes.

As we can see heart disease is mostly present in the average ages from or above 55 years, and the mean and distribution for no heart disease is below 55 years and has a wide distribution respectively.

Although outliers are present, heart disease usually is present in people in their late 30's.

7. Pairplot analysis



A subset of selected features, including 'age', 'trestbps', 'chol', 'thalach', 'oldpeak', and 'target', is extracted from the new_clev_DF DataFrame. The pairplot() function from seaborn is used to create a pairplot analysis of the selected features. The plot represents pairwise relationships between the features, with different target classes indicated by different colors. The diagonal plots show kernel density estimations (KDE) for each feature. This pairplot analysis helps identify relationships and patterns between the selected features.

Task 3 : Data Modelling

Do you think whether you can use logistic regression model to predict the heart disease? Explain.

Yes, I believe so. Logistic regression is a classification technique that uses supervised learning to estimate the likelihood of a target variable. It is a frequently used approach in predictive modeling for forecasting binary outcomes such as whether someone has heart disease or not. It uses a logistic function to connect features to a binary target variable. The purpose is to assess the likelihood that the target variable belongs to a specific class.

For this dataset for heart disease prediction, logistic regression is useful for determining how various characteristics, including as age, gender, cholesterol levels, and blood pressure, link to the chance of being diagnosed with heart disease. The model indicates the effect or contribution of each variable in predicting the likelihood of getting heart disease by calculating coefficients for each feature.

When it comes to predicting heart disease, logistic regression has several major advantages, including interpretability, flexibility, and predictive power.