Exploring the Heart Disease Dataset: An Analysis

IMPORT DATA

```
import pandas as pd  # Import Pandas library and alias it as 'pd'
```

Read data from csv file

```
Start coding or generate with AI.
```

```
data=pd.read_csv('/content/HeartDisease.csv')
```

```
data.shape
```

```
(303, 14)
```

Represents the dimensions of the data: 303 rows and 14 columns.

SHOW FIRST FEW ROWS

```
data.head(10)
```

	age	gender	chest_pain	rest_bps	cholestrol	fasting_blood_sugar	rest_ecg	thal
0	63	1	3	145	233	1	0	
1	37	1	2	130	250	0	1	
2	41	0	1	130	204	0	0	
3	56	1	1	120	236	0	1	
4	57	0	0	120	354	0	1	
5	57	1	0	140	192	0	1	
6	56	0	1	140	294	0	0	
7	44	1	1	120	263	0	1	
8	52	1	2	172	199	1	1	
9	57	1	2	150	168	0	1	

TO SHOW VARIABLE OR COLOUMN IN DATA

```
variable_name=data.columns
print("****Total Variable Name in DataFrame****")
print(variable_name)
```

```
****Total Variable Name in DataFrame****
Index(['age', 'gender', 'chest_pain', 'rest_bps', 'cholestrol',
       'fasting_blood_sugar', 'rest_ecg', 'thalach', 'exer_angina', 'old_peak',
       'slope', 'ca', 'thalassemia', 'target'],
      dtype='object')
```

DATA CLEANING

HANDLING MISSING VALUES

```
# Count missing values per column
Count_Missing_Value=data.isnull().sum()
print("\nNumber of missing values per column:")
print(Count_Missing_Value)
```

```
Number of missing values per column:
age                    0
gender                 0
chest_pain             0
rest_bps               0
cholestrol             0
fasting_blood_sugar    0
rest_ecg               0
thalach                0
exer_angina            0
old_peak               0
slope                  0
ca                     0
thalassemia            0
target                 0
dtype: int64
```

This analysis provides an overview of the completeness of data across different columns in the dataset, suggesting that there are no missing values present in any of the listed columns.

Thats why it is not required to do any kind of data cleaning step for removing or filling missing values

HANDLING DUPLICATE ROWS

```
# Count duplicate rows in the DataFrame
duplicate_count = data.duplicated().sum()

print("Number of duplicate rows in the DataFrame:", duplicate_count)
```

```
Number of duplicate rows in the DataFrame: 1
```

```
# Identify duplicate rows
print("Duplicate Rows:")
print(data[data.duplicated()])
```

```
Duplicate Rows:
     age  gender  chest_pain  rest_bps  cholestrol  fasting_blood_sugar  \
164   38       1           2       138         175                    0   

     rest_ecg  thalach  exer_angina  old_peak  slope  ca  thalassemia  target  
164         1      173            0       0.0      2   4            2       1  
```

Here only one dupliate row in dataframe

REMOVE DUPLICATE ROWS

```
data_unique = data.drop_duplicates()

print("\nDataFrame after removing duplicates:")
print(data_unique)
```

```
DataFrame after removing duplicates:
     age  gender  chest_pain  rest_bps  cholestrol  fasting_blood_sugar  \
0     63       1           3       145         233                    1   
1     37       1           2       130         250                    0   
2     41       0           1       130         204                    0   
3     56       1           1       120         236                    0   
4     57       0           0       120         354                    0   
..   ...     ...         ...       ...         ...                  ...   
298   57       0           0       140         241                    0   
299   45       1           3       110         264                    0   
300   68       1           0       144         193                    1   
301   57       1           0       130         131                    0   
302   57       0           1       130         236                    0   

     rest_ecg  thalach  exer_angina  old_peak  slope  ca  thalassemia  target  
0           0      150            0       2.3      0   0            1       1  
1           1      187            0       3.5      0   0            2       1  
2           0      172            0       1.4      2   0            2       1  
3           1      178            0       0.8      2   0            2       1  
4           1      163            1       0.6      2   0            2       1  
```

```
..       ...     ...         ...      ...    ...  ..          ...     ...
298        1     123           1      0.2      1   0            3       0
299        1     132           0      1.2      1   0            3       0
300        1     141           0      3.4      1   2            3       0
301        1     115           1      1.2      1   1            3       0
302        0     174           0      0.0      1   1            2       0

[302 rows x 14 columns]
```

DESCRIPTIVE ANALYSIS

```
data.describe()
```

	age	gender	chest_pain	rest_bps	cholestrol	fasting_blood_sugar
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000

The dataset presents a demographic snapshot of individuals at risk of heart disease, showcasing key physiological markers. On average, individuals are around 54 years old, with ages spanning from 29 to 77 years. Resting blood pressure averages at 132 mmHg, fluctuating between 94 mmHg and 200 mmHg across individuals. Cholesterol levels vary widely, with an average of 246 mg/dL and values ranging from 126 mg/dL to 564 mg/dL. These statistics underscore the heterogeneity in cardiovascular health metrics within the dataset, laying the groundwork for deeper analysis into risk factors and disease prevalence.

MEAN OF EACH VARIABLE

```
Mean=data.mean()
print("Mean of each Variable")
print (Mean)
```

```
Mean of each Variable
age                     54.366337
gender                   0.683168
chest_pain               0.966997
rest_bps               131.623762
cholestrol             246.264026
fasting_blood_sugar      0.148515
rest_ecg                 0.528053
thalach                149.646865
exer_angina              0.326733
old_peak                 1.039604
slope                    1.399340
ca                       0.729373
thalassemia              2.313531
target                   0.544554
dtype: float64
```

MEDIAN OF EACH VARIABLE

```
Median=data.median()
print("Median of each Variable")
print (Median)
```

```
Median of each Variable
age                     55.0
gender                   1.0
chest_pain               1.0
rest_bps               130.0
cholestrol             240.0
fasting_blood_sugar      0.0
rest_ecg                 1.0
thalach                153.0
exer_angina              0.0
old_peak                 0.8
slope                    1.0
```

```
ca                        0.0
thalassemia               2.0
target                    1.0
dtype: float64
```

MODE OF EACH VARIABLE

```
Mode=data.mean()
print("Mode of each Variable:")
print(Mode)
```

```
Mode of each Variable:
age                      54.366337
gender                    0.683168
chest_pain                0.966997
rest_bps                131.623762
cholestrol              246.264026
fasting_blood_sugar       0.148515
rest_ecg                  0.528053
thalach                 149.646865
exer_angina               0.326733
old_peak                  1.039604
slope                     1.399340
ca                        0.729373
thalassemia               2.313531
target                    0.544554
dtype: float64
```

CORRECTING INCONSISTENT DATA

```
data['gender'].replace(0,'Male',inplace=True)
data['gender'].replace(1,'Female',inplace=True)
```

```
data.head(10)
```

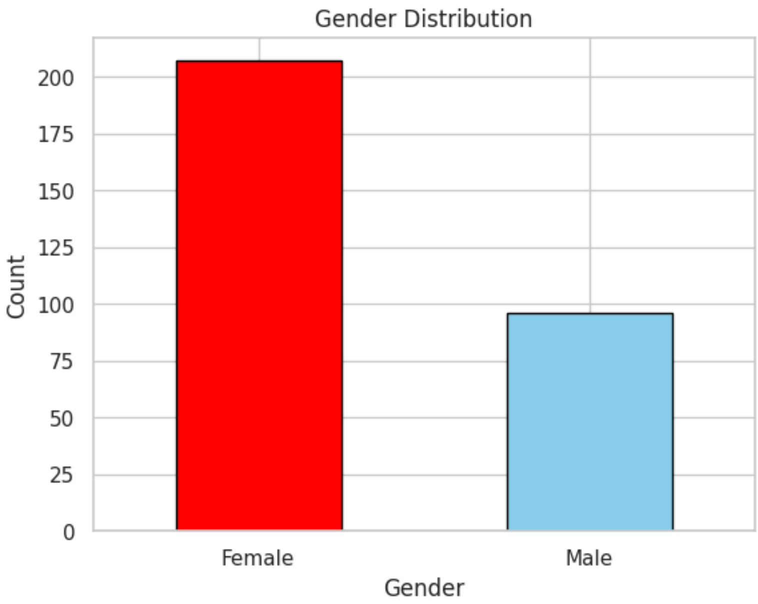
	age	gender	chest_pain	rest_bps	cholestrol	fasting_blood_sugar	rest_ecg	thal
0	63	Female	3	145	233	1	0	
1	37	Female	2	130	250	0	1	
2	41	Male	1	130	204	0	0	
3	56	Female	1	120	236	0	1	
4	57	Male	0	120	354	0	1	
5	57	Female	0	140	192	0	1	
6	56	Male	1	140	294	0	0	
7	44	Female	1	120	263	0	1	
8	52	Female	2	172	199	1	1	
9	57	Female	2	150	168	0	1	

DATA VISUALIZATION

```
import matplotlib.pyplot as plt
import seaborn as sns

# Set up the Seaborn style
sns.set(style="whitegrid")
```

GENDER DISTRIBUTION

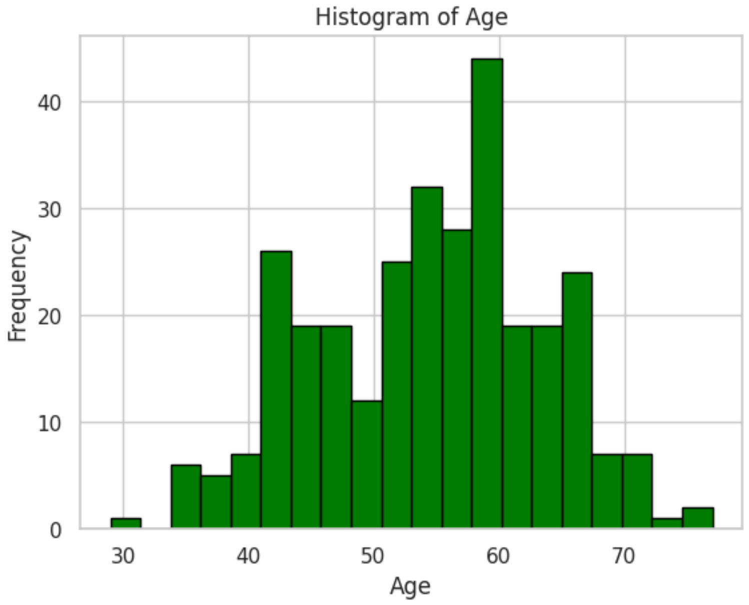
```
# Calculate gender counts
gender_counts = data['gender'].value_counts()

# Define colors for male and female
colors = ['red', 'skyblue']  # Reversed colors according to the reversed gender representation

# Create bar chart with specified colors
gender_counts.plot(kind='bar', color=colors, edgecolor='black')

# Add labels and title
plt.xlabel('Gender')
plt.ylabel('Count')
plt.title('Gender Distribution')

# Customize x-axis ticks and labels
plt.xticks([0, 1], ['Female', 'Male'], rotation=0)  # Reversed the labels

# Show plot
plt.show()
```



The histogram illustrates gender distribution in a heart disease analysis dataset, showing 207 females and 96 males. This suggests a potential gender discrepancy in the prevalence or risk factors associated with heart disease.

HISTOGRAM OF AGE

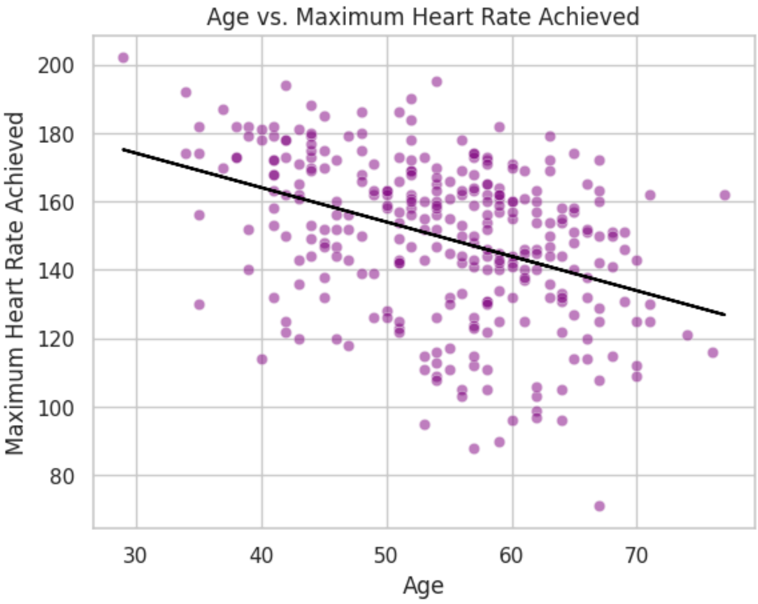
```
plt.hist(data['age'], bins=20, color='green', edgecolor='black')
plt.title('Histogram of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```



The dataset on heart disease encompasses individuals ranging from 29 to 77 years old, with a mean age of approximately 54.4 years. Notably, the oldest person recorded is 77 years old, while the youngest is 29.

PLOT AGE VS. MAXIMUM HEART RATE ACHIEVED AS A SCATTER PLOT

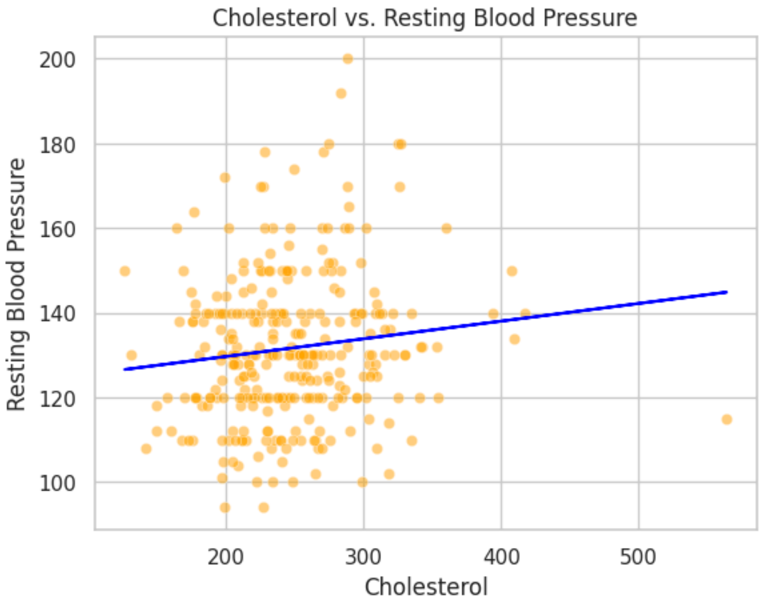
```
sns.scatterplot(data=data, x='age', y='thalach', color='purple', alpha=0.5)

# Calculate the trend line
x = data['age']
y = data['thalach']
m, b = np.polyfit(x, y, 1)
plt.plot(x, m*x + b, color='black')

plt.title('Age vs. Maximum Heart Rate Achieved')
plt.xlabel('Age')
plt.ylabel('Maximum Heart Rate Achieved')

plt.show()
```



The scatter plot of age versus maximum heart rate achieved shows a simple trend: as people get older, their maximum heart rate tends to decrease. This straightforward observation emphasizes how age influences heart health, with advancing age typically correlating with lower maximum heart rates.

CHOLESTEROL VS. RESTING BLOOD PRESSURE AS A SCATTER PLOT

```
sns.scatterplot(data=data, x='cholestrol', y='rest_bps', color='orange', alpha=0.5)

# Calculate the trend line
x = data['cholestrol']
y = data['rest_bps']
m, b = np.polyfit(x, y, 1)
plt.plot(x, m*x + b, color='blue')

plt.title('Cholesterol vs. Resting Blood Pressure')
plt.xlabel('Cholesterol')
plt.ylabel('Resting Blood Pressure')

plt.show()
```

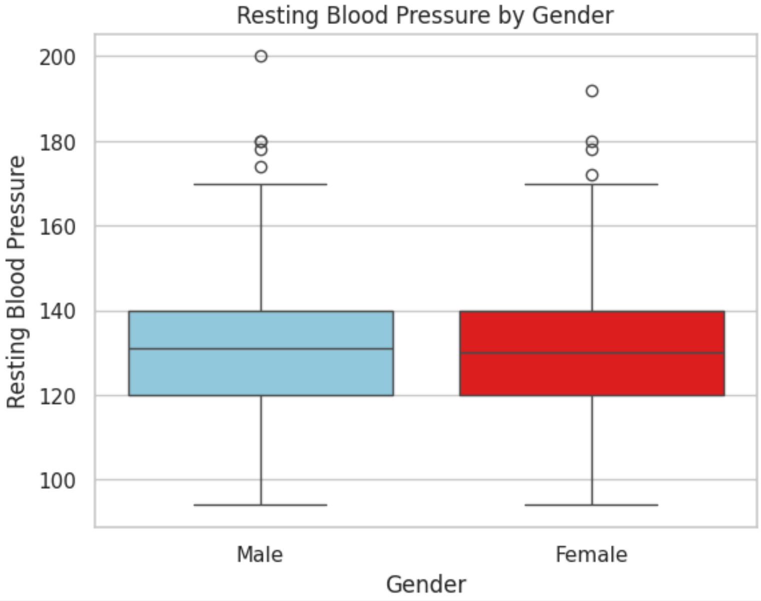


The scatter plot of cholesterol versus resting blood pressure reveals a clear upward trend, indicating a positive correlation between these variables. This suggests that as cholesterol levels increase, resting blood pressure tends to rise. This observation underscores the potential association between cholesterol levels and hypertension, essential factors in assessing cardiovascular health.

BOX PLOT OF RESTING BLOOD PRESSURE BY GENDE

```
sns.boxplot(x='gender', y='rest_bps', data=data, palette=['skyblue', 'red'])

plt.title('Resting Blood Pressure by Gender')
plt.xlabel('Gender')
plt.ylabel('Resting Blood Pressure')

plt.xticks(ticks=[0, 1], labels=['Male', 'Female'])  # Specify x-axis labels

plt.show()
```

```
<ipython-input-92-674e94edeb5f>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.

  sns.boxplot(x='gender', y='rest_bps', data=data, palette=['skyblue', 'red'])
```



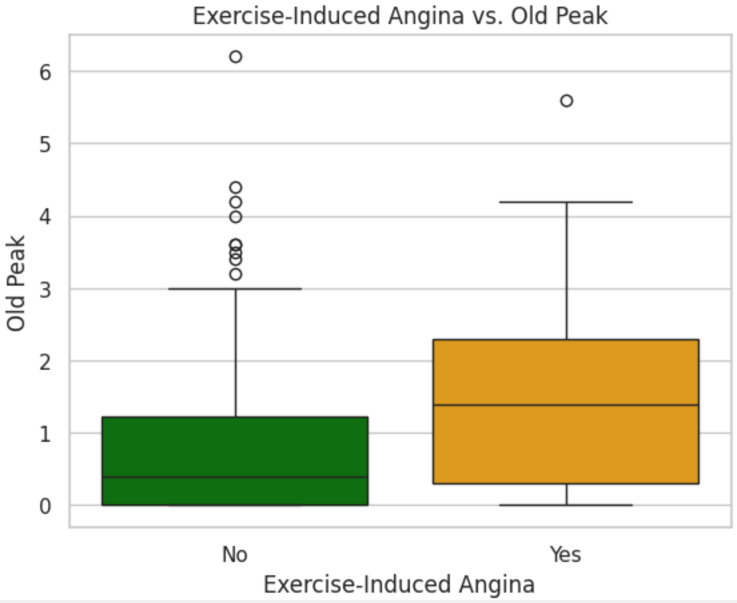
The box plot for gender against resting blood pressure a median value for male and female is 120around and 140s, with a number of outliers on the higher end.This suggests that while the majority of individuals have resting blood pressure within an expected range, there are a few with significantly high levels

---

BOX PLOT OF EXERCISE-INDUCED ANGINA VS. OLD PEAK

```
sns.boxplot(x='exer_angina', y='old_peak', data=data, palette=['green', 'orange'])

plt.title('Exercise-Induced Angina vs. Old Peak')
plt.xlabel('Exercise-Induced Angina')
plt.ylabel('Old Peak')

plt.xticks(ticks=[0, 1], labels=['No', 'Yes'])  # Specify x-axis labels

plt.show()
```

```
<ipython-input-82-a379db26d8c9>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.boxplot(x='exer_angina', y='old_peak', data=data, palette=['green', 'orange'])
```



The box plot shows that individuals with exercise-induced angina typically have old peak values ranging from 0.2 to 3, while those without it tend to have values between 0 and 1. This suggests that higher old peak values are associated with exercise-induced angina, providing a straightforward indicator of its occurrence during physical activity.

CORRELATION ANALYSIS

# COMPUTE CORRELATION MATRIX

```
# Replace 'Male' with 0 and 'Female' with 1
# Reason: We need to convert categorical values ('Male' and 'Female') into numerical values (0 and 1)
# to calculate correlation matrix.
data['gender'].replace("Male", 0, inplace=True)
data['gender'].replace('Female', 1, inplace=True)

# Compute correlation matrix
correlation_matrix = data.corr()

# Add title
print("**Correlation Matrix:**")

# Display correlation matrix
print(correlation_matrix)
```

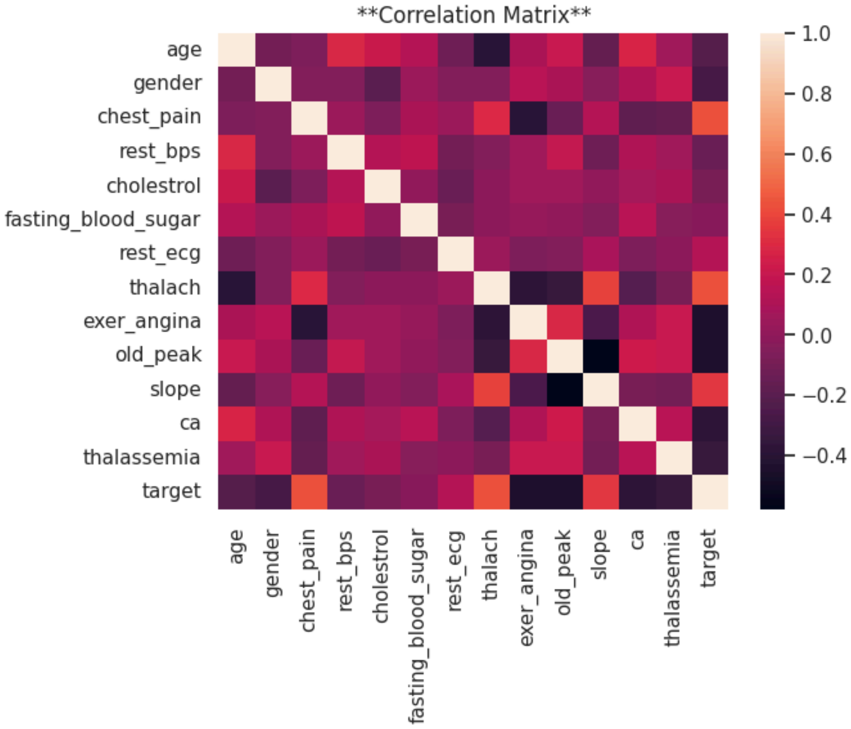
```
**Correlation Matrix:**
                          age    gender  chest_pain  rest_bps  cholestrol  \
age                  1.000000 -0.098447   -0.068653  0.279351    0.213678
gender              -0.098447  1.000000   -0.049353 -0.056769   -0.197912
chest_pain          -0.068653 -0.049353    1.000000  0.047608   -0.076904
rest_bps             0.279351 -0.056769    0.047608  1.000000    0.123174
cholestrol           0.213678 -0.197912   -0.076904  0.123174    1.000000
fasting_blood_sugar  0.121308  0.045032    0.094444  0.177531    0.013294
rest_ecg            -0.116211 -0.058196    0.044421 -0.114103   -0.151040
thalach             -0.398522 -0.044020    0.295762 -0.046698   -0.009940
exer_angina          0.096801  0.141664   -0.394280  0.067616    0.067023
old_peak             0.210013  0.096093   -0.149230  0.193216    0.053952
slope               -0.168814 -0.030711    0.119717 -0.121475   -0.004038
ca                   0.276326  0.118261   -0.181053  0.101389    0.070511
thalassemia          0.068001  0.210041   -0.161736  0.062210    0.098803
target              -0.225439 -0.280937    0.433798 -0.144931   -0.085239

                     fasting_blood_sugar  rest_ecg   thalach  exer_angina  \
age                             0.121308 -0.116211 -0.398522     0.096801
gender                          0.045032 -0.058196 -0.044020     0.141664
chest_pain                      0.094444  0.044421  0.295762    -0.394280
rest_bps                        0.177531 -0.114103 -0.046698     0.067616
cholestrol                      0.013294 -0.151040 -0.009940     0.067023
fasting_blood_sugar             1.000000 -0.084189 -0.008567     0.025665
rest_ecg                       -0.084189  1.000000  0.044123    -0.070733
thalach                        -0.008567  0.044123  1.000000    -0.378812
exer_angina                     0.025665 -0.070733 -0.378812     1.000000
old_peak                        0.005747 -0.058770 -0.344187     0.288223
slope                          -0.059894  0.093045  0.386784    -0.257748
ca                              0.137979 -0.072042 -0.213177     0.115739
thalassemia                    -0.032019 -0.011981 -0.096439     0.206754
target                         -0.028046  0.137230  0.421741    -0.436757

                     old_peak     slope        ca  thalassemia    target
age                  0.210013 -0.168814  0.276326     0.068001 -0.225439
gender               0.096093 -0.030711  0.118261     0.210041 -0.280937
chest_pain          -0.149230  0.119717 -0.181053    -0.161736  0.433798
rest_bps             0.193216 -0.121475  0.101389     0.062210 -0.144931
cholestrol           0.053952 -0.004038  0.070511     0.098803 -0.085239
fasting_blood_sugar  0.005747 -0.059894  0.137979    -0.032019 -0.028046
rest_ecg            -0.058770  0.093045 -0.072042    -0.011981  0.137230
thalach             -0.344187  0.386784 -0.213177    -0.096439  0.421741
exer_angina          0.288223 -0.257748  0.115739     0.206754 -0.436757
old_peak             1.000000 -0.577537  0.222682     0.210244 -0.430696
slope               -0.577537  1.000000 -0.080155    -0.104764  0.345877
ca                   0.222682 -0.080155  1.000000     0.151832 -0.391724
thalassemia          0.210244 -0.104764  0.151832     1.000000 -0.344029
target              -0.430696  0.345877 -0.391724    -0.344029  1.000000
```

HEAT MAP

This heatmap visually encapsulates the relationships between variables, where values close to 1 represent strong positive correlations, values near -1 indicate strong negative correlations, and values around 0 signify weak or no correlation.

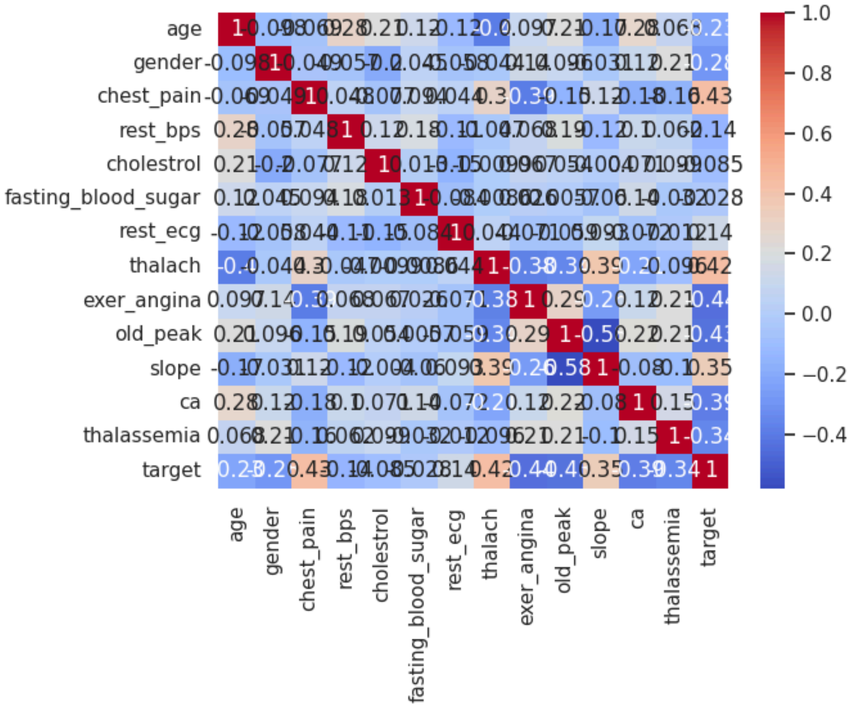
```
# Create heatmap of correlation matrix
heatmap = sns.heatmap(data.corr())

# Add title to the heatmap
heatmap.set_title('**Correlation Matrix**')
```

```
Text(0.5, 1.0, '**Correlation Matrix**')
```



```
correlation_matrix=data.corr()
sns.heatmap(correlation_matrix,annot=True,cmap='coolwarm')
```

```
<Axes: >
```



FINDING: ***Strong Positive Correlations:***

> A chest_pain' and 'target' have a correlation coefficient of 0.433798, indicating a strong positive correlation. This suggests that as chest pain severity increases, the likelihood of having heart disease (target) also increases. 'thalach' and 'target' have a correlation coefficient of 0.421741, indicating a strong positive correlation. This implies that as maximum heart rate achieved during exercise ('thalach') increases, the likelihood of having heart disease (target) also increases.dd blockquote

Weak Correlations:

> gender' and 'rest_bps' have a correlation coefficient of -0.056769, indicating a weak negative correlation. This suggests a slight tendency for gender to be associated with lower resting blood pressure ('rest_bps'). 'age' and 'thalach' have a correlation coefficient of -0.398522, indicating a moderate negative correlation. This suggests that as age increases, the maximum heart rate achieved during exercise ('thalach') tends to decrease, albeit not very strongly.

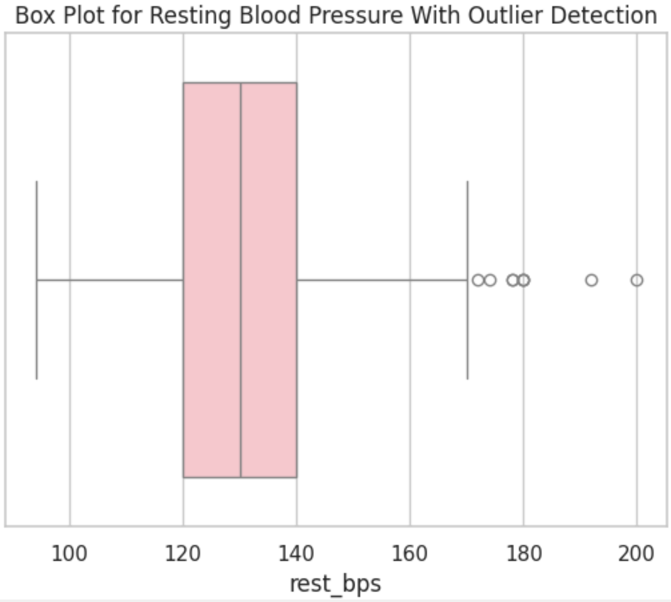
'

OUTLIER DETECTION

BOX PLOT FOR RESTING BLOOD PRESSURE WITH OUTLIER DETECTION

```
Outlier_Detection = sns.boxplot(x=data['rest_bps'], palette=['pink'])
Outlier_Detection.set_title("Box Plot for Resting Blood Pressure With Outlier Detection")
plt.show()
```

```
<ipython-input-86-45dc1f692faa>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.

  Outlier_Detection = sns.boxplot(x=data['rest_bps'], palette=['pink'])
```



This box plot depicts the variability in resting blood pressure among individuals in the dataset. While the median value stands at around 130 mmHg, outliers exist at both ends, suggesting notable deviations in blood pressure readings across the sample.

IDENTIFYING CHOLESTEROL OUTLIERS USING Z-SCORE ANALYSIS"

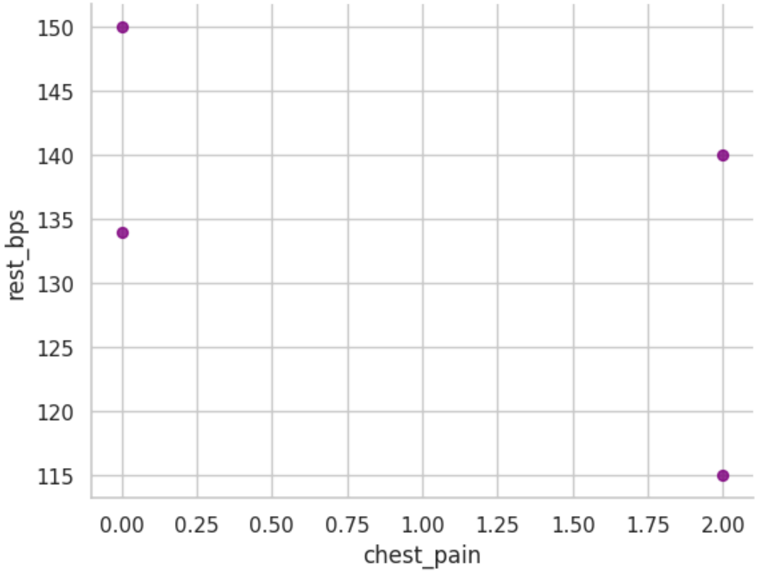
```
from scipy.stats import zscore

# Calculating Z-scores for cholesterol data
data['z_score'] = zscore(data['cholestrol'])

# Identifying outliers based on Z-score
outliers = data[(data['z_score'] > 3) | (data['z_score'] < -3)]

# Displaying outliers
print("Outliers Detected Using Z-Score Analysis:")
print(outliers)
```

```
Outliers Detected Using Z-Score Analysis:
     age  gender  chest_pain  rest_bps  cholestrol  fasting_blood_sugar  \
28    65       0           2       140         417                    1
85    67       0           2       115         564                    0
220   63       0           0       150         407                    0
246   56       0           0       134         409                    0

     rest_ecg  thalach  exer_angina  old_peak  slope  ca  thalassemia  target  \
28          0      157            0       0.8      2   1            2       1
85          0      160            0       1.6      1   0            3       1
220         0      154            0       4.0      1   3            3       0
246         0      150            1       1.9      1   2            3       0

      z_score
28   3.299555
85   6.140401
220  3.106300
246  3.144951
```

In the DataFrame, it is evident that only one 'cholestrol' value exhibits a z-score exceeding 3, indicating a high likelihood of it being an outlier.

SCATTER PLOT FOR OUTLIERS

```
outliers.plot(kind='scatter', x='chest_pain', y='rest_bps', s=32, alpha=0.8 , color='purple')

# Remove top and right spines
plt.gca().spines['top'].set_visible(False)
plt.gca().spines['right'].set_visible(False)

# Show the plot
plt.show()
```



LOG TRANSFORMATION

```
import numpy as np
data['log_transformed'] = np.log(data['age'])

# Standardization
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
data['standardized_column'] = scaler.fit_transform(data['age'].values.reshape(-1, 1))
data['standardized_column']
```

```
0      0.952197
1     -1.915313
2     -1.474158
3      0.180175
4      0.290464
```