```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

PREPROCESSING

```
columns = ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang',  'oldpeak', 'slope', 'ca', 'thal', 'targe
```

```
df=pd.read_csv('processed.cleveland.data',names=columns)
```

```
df.head(7)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca
0	63.0	1.0	1.0	145.0	233.0	1.0	2.0	150.0	0.0	2.3	3.0	0.0
1	67.0	1.0	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	2.0	3.0
2	67.0	1.0	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	2.0	2.0
3	37.0	1.0	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	3.0	0.0
4	41.0	0.0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	1.0	0.0
5	56.0	1.0	2.0	120.0	236.0	0.0	0.0	178.0	0.0	0.8	1.0	0.0
6	62.0	0.0	4.0	140.0	268.0	0.0	2.0	160.0	0.0	3.6	3.0	2.0

```
df.dtypes
```

```
age          float64
sex          float64
cp           float64
trestbps     float64
chol         float64
fbs          float64
restecg      float64
thalach      float64
exang        float64
oldpeak      float64
slope        float64
ca            object
thal          object
target         int64
dtype: object
```

```
df.isnull().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

```
df['sex'].unique()
```

```
array([1., 0.])
```

```
df['thal'] = pd.to_numeric(df['thal'], errors='coerce')
df['thal'] = df['thal'].fillna(0)
df['thal'] = df['thal'].astype(int)

df['ca'] = pd.to_numeric(df['ca'], errors='coerce')
df['ca'] = df['ca'].fillna(0)
df['ca'] = df['ca'].astype(int)
df['sex'] = df['sex'].astype(int)
```

```
df.dtypes
```

```
age         float64
sex           int32
cp          float64
trestbps    float64
chol        float64
fbs         float64
restecg     float64
thalach     float64
exang       float64
oldpeak     float64
slope       float64
ca            int32
thal          int32
target        int64
dtype: object
```

```
df.head(10)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca
0	63.0	1	1.0	145.0	233.0	1.0	2.0	150.0	0.0	2.3	3.0	0
1	67.0	1	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	2.0	3
2	67.0	1	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	2.0	2
3	37.0	1	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	3.0	0
4	41.0	0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	1.0	0
5	56.0	1	2.0	120.0	236.0	0.0	0.0	178.0	0.0	0.8	1.0	0
6	62.0	0	4.0	140.0	268.0	0.0	2.0	160.0	0.0	3.6	3.0	2
7	57.0	0	4.0	120.0	354.0	0.0	0.0	163.0	1.0	0.6	1.0	0
8	63.0	1	4.0	130.0	254.0	0.0	2.0	147.0	0.0	1.4	2.0	1
9	53.0	1	4.0	140.0	203.0	1.0	2.0	155.0	1.0	3.1	3.0	0

```
df['target'].value_counts()
```

```
target
0    164
1     55
2     36
3     35
4     13
Name: count, dtype: int64
```

```
df['target']=df['target'].apply(lambda x: 1 if x>1 else x)
```

```
df['target'].value_counts()
```

```
target
0    164
1    139
Name: count, dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       303 non-null    float64
 1   sex       303 non-null    int32  
 2   cp        303 non-null    float64
 3   trestbps  303 non-null    float64
 4   chol      303 non-null    float64
 5   fbs       303 non-null    float64
 6   restecg   303 non-null    float64
 7   thalach   303 non-null    float64
 8   exang     303 non-null    float64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    float64
 11  ca        303 non-null    int32  
 12  thal      303 non-null    int32  
 13  target    303 non-null    int64  
dtypes: float64(10), int32(3), int64(1)
memory usage: 29.7 KB
```

Descriptive Statistics

```
df.describe()
```

	age	sex	cp	trestbps	chol	fbs	reste
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0000
mean	54.438944	0.679868	3.158416	131.689769	246.693069	0.148515	0.9900
std	9.038662	0.467299	0.960126	17.599748	51.776918	0.356198	0.994
min	29.000000	0.000000	1.000000	94.000000	126.000000	0.000000	0.0000
25%	48.000000	0.000000	3.000000	120.000000	211.000000	0.000000	0.0000
50%	56.000000	1.000000	3.000000	130.000000	241.000000	0.000000	1.0000
75%	61.000000	1.000000	4.000000	140.000000	275.000000	0.000000	2.0000
max	77.000000	1.000000	4.000000	200.000000	564.000000	1.000000	2.0000

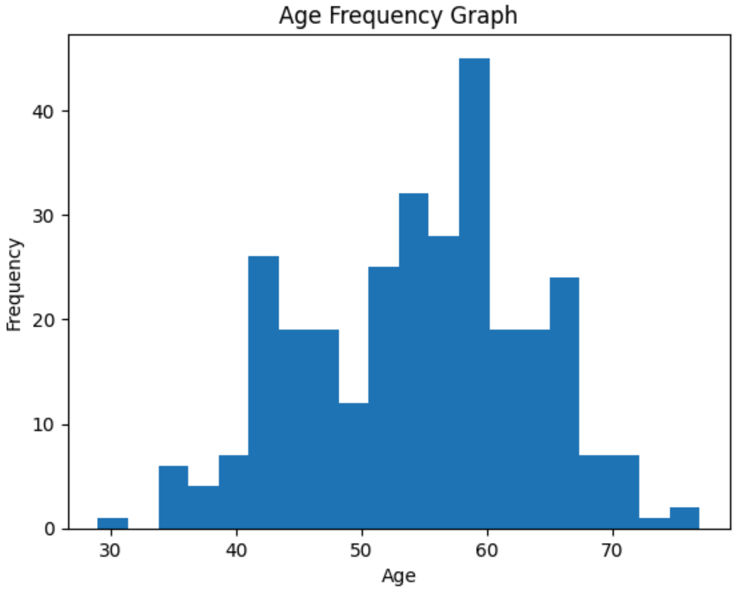
TASKS

Task 1: Age Distribution Create a histogram to visualize the distribution of ages in the dataset. • Use Matplotlib to create the histogram. • Label the axes and provide a title.

```
df['age'].value_counts()
```

```
age
58.0    19
57.0    17
54.0    16
59.0    14
52.0    13
60.0    12
51.0    12
56.0    11
62.0    11
44.0    11
64.0    10
41.0    10
67.0     9
63.0     9
42.0     8
43.0     8
45.0     8
53.0     8
55.0     8
61.0     8
65.0     8
50.0     7
66.0     7
48.0     7
46.0     7
47.0     5
49.0     5
70.0     4
68.0     4
35.0     4
39.0     4
69.0     3
71.0     3
40.0     3
34.0     2
37.0     2
38.0     2
29.0     1
77.0     1
74.0     1
76.0     1
Name: count, dtype: int64
```

```
#Age frequency graph
plt.hist(df['age'],bins=20)
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title("Age Frequency Graph")
plt.show()
```



Start coding or generate with AI.

Task 2: Gender Distribution Create a bar plot to visualize the distribution of gender in the dataset. • Use Seaborn to create the bar plot. • Label the axes and provide a title.

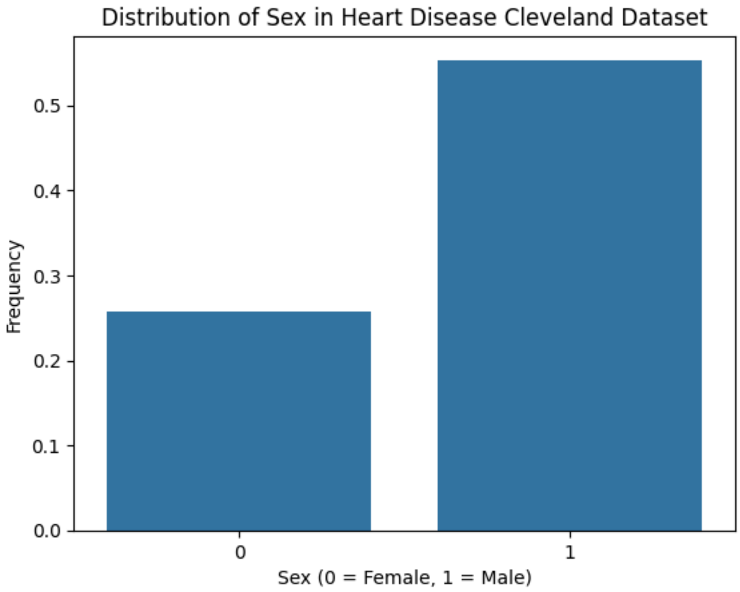
```
df.groupby('sex').size().reset_index(name='count')
```

	sex	count
0	0	97
1	1	206

```
sns.barplot(x='sex',y='target', data=df,errorbar=None)

plt.title('Distribution of Sex in Heart Disease Cleveland Dataset')
plt.xlabel('Sex (0 = Female, 1 = Male)')
plt.ylabel('Frequency')

plt.show()
```

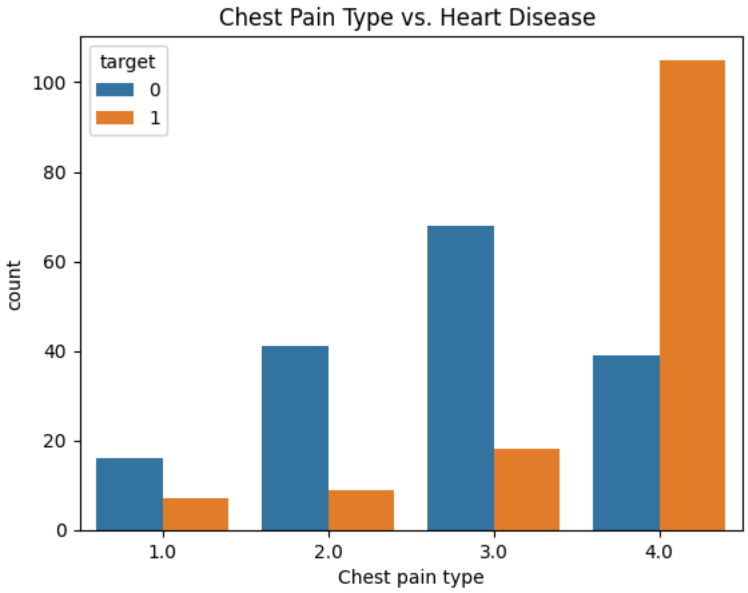


Start coding or generate with AI.

Task 3: Chest Pain Type vs. Heart Disease Create a count plot to visualize the relationship between chest pain type (cp) and the presence of heart disease (target). • Use Seaborn to create the count plot. • Use different colors to differentiate between the presence and absence of heart disease. • Label the axes and provide a title.

```
sns.countplot(data=df,x='cp',hue='target')
plt.xlabel('Chest pain type')
plt.title('Chest Pain Type vs. Heart Disease ')
```

```
Text(0.5, 1.0, 'Chest Pain Type vs. Heart Disease ')
```



Start coding or generate with AI.

Task 4: Cholesterol Levels Create a box plot to visualize the distribution of cholesterol levels (chol) for patients with and without heart disease. • Use Seaborn to create the box plot. • Label the axes and provide a title.

```
df
```

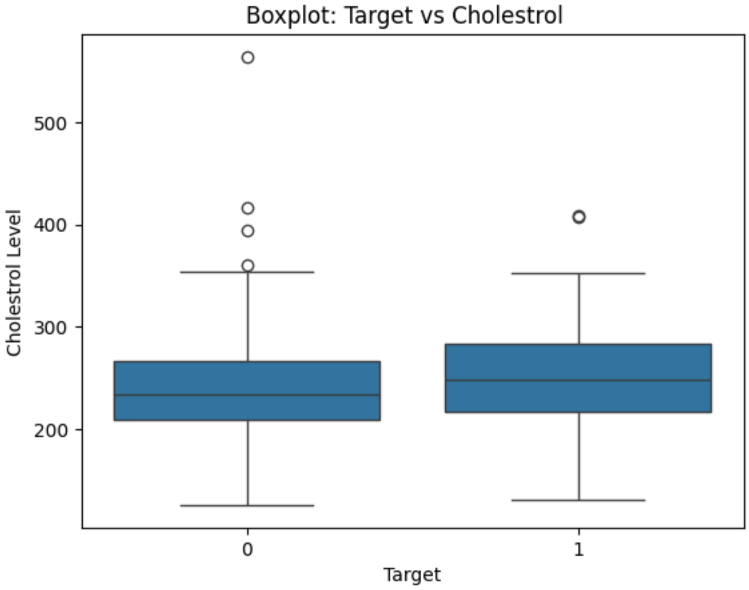
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	c
0	63.0	1	1.0	145.0	233.0	1.0	2.0	150.0	0.0	2.3	3.0	
1	67.0	1	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	2.0	
2	67.0	1	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	2.0	
3	37.0	1	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	3.0	
4	41.0	0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	1.0	
...	...	...	...	...	...	...	...	...	...	...	...	.
298	45.0	1	1.0	110.0	264.0	0.0	0.0	132.0	0.0	1.2	2.0	
299	68.0	1	4.0	144.0	193.0	1.0	0.0	141.0	0.0	3.4	2.0	
300	57.0	1	4.0	130.0	131.0	0.0	0.0	115.0	1.0	1.2	2.0	
301	57.0	0	2.0	130.0	236.0	0.0	2.0	174.0	0.0	0.0	2.0	
302	38.0	1	3.0	138.0	175.0	0.0	0.0	173.0	0.0	0.0	1.0	

303 rows × 14 columns

```
sns.boxplot(data=df,y='chol',x='target')

plt.xlabel('Target')
plt.ylabel('Cholestrol Level')
plt.title("Boxplot: Target vs Cholestrol  ")
```

```
Text(0.5, 1.0, 'Boxplot: Target vs Cholestrol  ')
```



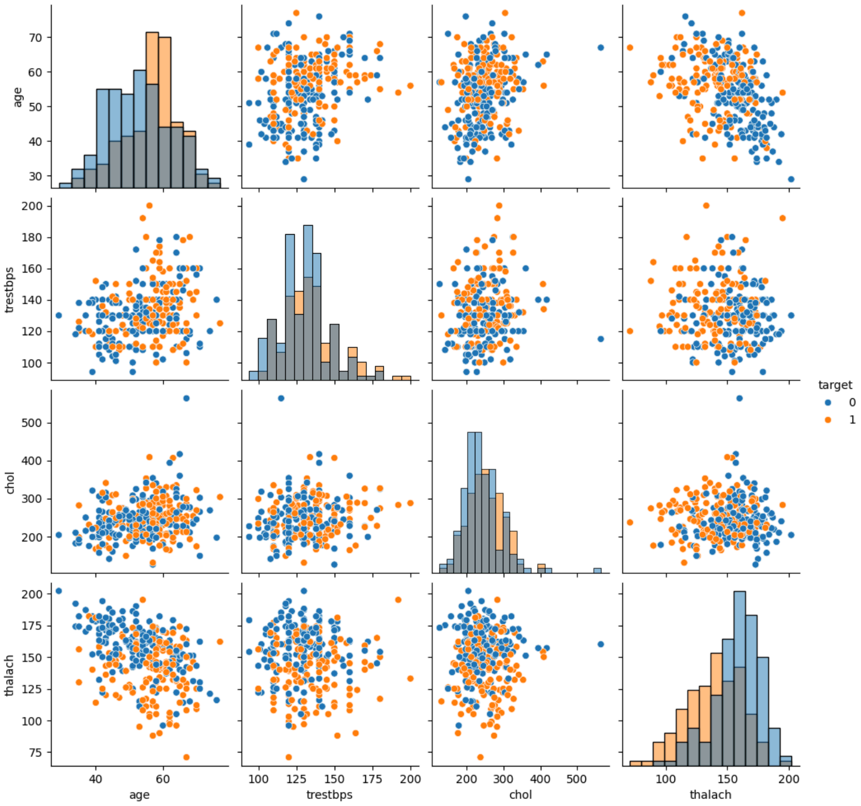
Start coding or generate with AI.

Task 5: Pair Plot Create a pair plot to visualize relationships between multiple variables. • Use Seaborn to create the pair plot. • Include the following variables: age, trestbps, chol, thalach, and target. • Differentiate the points based on the target variable.

```
selected_columns = ['age', 'trestbps', 'chol', 'thalach', 'target']
df_selected = df[selected_columns]

sns.pairplot(df_selected, hue='target', diag_kind='hist')
```

```
<seaborn.axisgrid.PairGrid at 0x24f32ffda90>
```

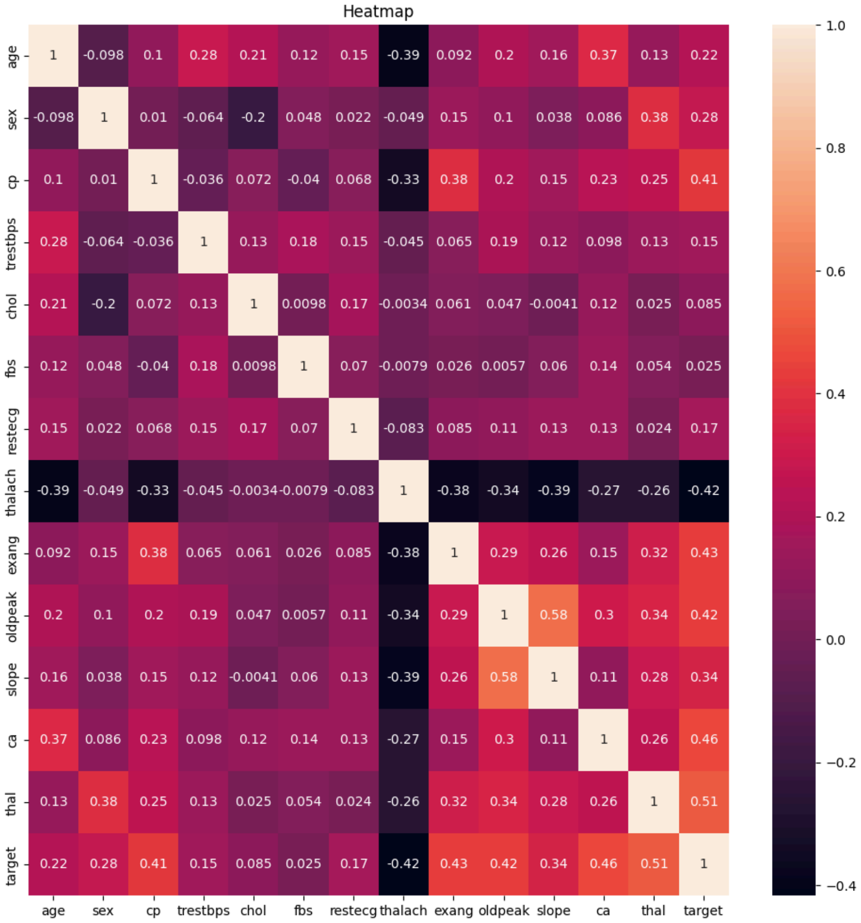


Start coding or generate with AI.

Task 6: Correlation Heatmap Create a heatmap to visualize the correlation between different attributes in the dataset. • Use Seaborn to create the heatmap. • Display the correlation values on the heatmap. • Provide a title.

```
corr_rel=df.corr()
plt.figure(figsize=(12,12))
sns.heatmap(corr_rel,annot=True)
plt.title ('Heatmap')
```

```
Text(0.5, 1.0, 'Heatmap')
```



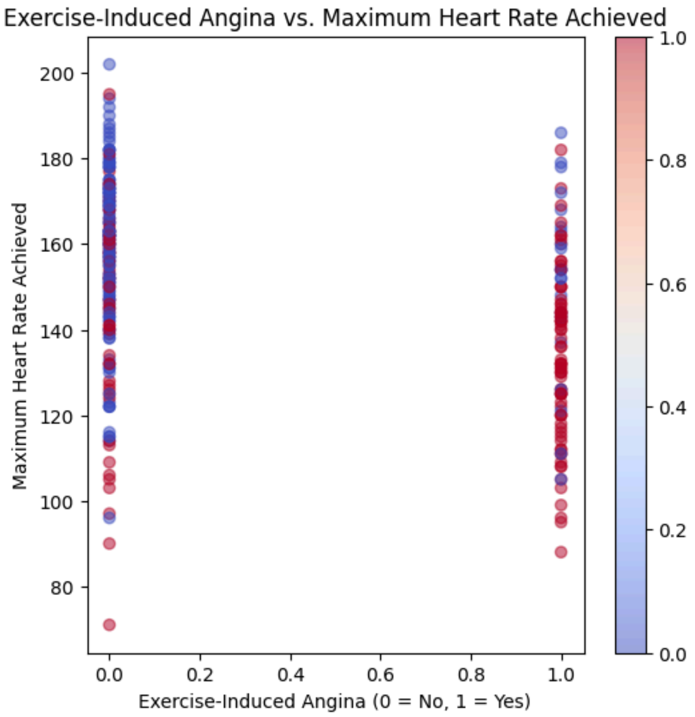
Task 7: Exercise Induced Angina vs. Maximum Heart Rate Create a scatter plot to visualize the relationship between exercise-induced angina (exang) and maximum heart rate (thalach). • Use Matplotlib to create the scatter plot. • Color the points based on the presence of heart disease (target). • Label the axes and provide a title.

```
plt.figure(figsize=(6, 6))
scatter = plt.scatter(df['exang'], df['thalach'], c=df['target'], cmap='coolwarm', alpha=0.5)
```

```
plt.figure(figsize=(6, 6))
scatter = plt.scatter(df['exang'], df['thalach'], c=df['target'], cmap='coolwarm', alpha=0.5)

plt.title('Exercise-Induced Angina vs. Maximum Heart Rate Achieved')
plt.xlabel('Exercise-Induced Angina (0 = No, 1 = Yes)')
plt.ylabel('Maximum Heart Rate Achieved')

cbar = plt.colorbar(scatter)
```



Start coding or generate with AI.