ai-ml-tasks

June 5, 2025

https://colab.research.google.com/drive/1SR0yLmwUS68UMHsDGHSB1Ofzt_kNPzYY?usp=sharing

1 Github Repository

[61]:
```
%cd https://github.com/zoya4477/AI-Ml.git
!git clone
!git config --global user.email "zoyahafeez785@gmail.com"
!git config --global user.name "zoya4477"
```

```
[Errno 2] No such file or directory: 'https://github.com/zoya4477/AI-Ml.git'
/content
fatal: You must specify a repository to clone.

usage: git clone [<options>] [--] <repo> [<dir>]

    -v, --verbose         be more verbose
    -q, --quiet           be more quiet
    --progress            force progress reporting
    --reject-shallow      don't clone shallow repository
    -n, --no-checkout     don't create a checkout
    --bare                create a bare repository
    --mirror              create a mirror repository (implies bare)
    -l, --local           to clone from a local repository
    --no-hardlinks        don't use local hardlinks, always copy
    -s, --shared          setup as shared repository
    --recurse-submodules[=<pathspec>]
                          initialize submodules in the clone
    --recursive …       alias of --recurse-submodules
    -j, --jobs <n>        number of submodules cloned in parallel
    --template <template-directory>
                          directory from which templates will be used
    --reference <repo>    reference repository
    --reference-if-able <repo>
                          reference repository
    --dissociate          use --reference only while cloning
    -o, --origin <name>   use <name> instead of 'origin' to track upstream
    -b, --branch <branch>
```

```
                          checkout <branch> instead of the remote's HEAD
    -u, --upload-pack <path>
                          path to git-upload-pack on the remote
    --depth <depth>       create a shallow clone of that depth
    --shallow-since <time>
                          create a shallow clone since a specific time
    --shallow-exclude <revision>
                          deepen history of shallow clone, excluding rev
    --single-branch       clone only one branch, HEAD or --branch
    --no-tags             don't clone any tags, and make later fetches not to
follow them
    --shallow-submodules  any cloned submodules will be shallow
    --separate-git-dir <gitdir>
                          separate git dir from working tree
    -c, --config <key=value>
                          set config inside the new repository
    --server-option <server-specific>
                          option to transmit
    -4, --ipv4            use IPv4 addresses only
    -6, --ipv6            use IPv6 addresses only
    --filter <args>       object filtering
    --remote-submodules   any cloned submodules will use their remote-tracking
branch
    --sparse              initialize sparse-checkout file to include only files
at root
```

2 Task 1: Exploring and Visualizing the Iris Dataset

3 Load the Dataset

[30]:
```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the Iris dataset directly from seaborn
df = pd.read_csv('/content/Iris.csv')
```

4 Inspect the dataset

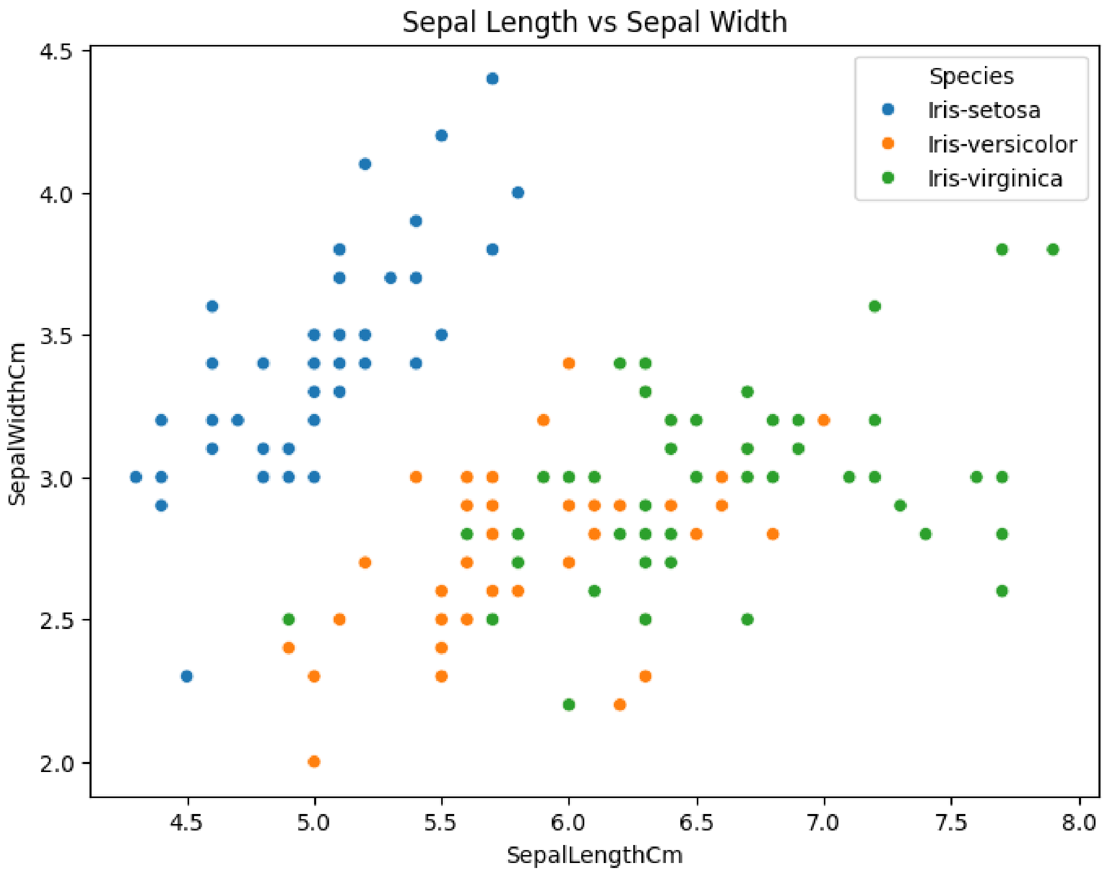
[31]:
```python
# Shape of the dataset
print("Shape of the dataset:", df.shape)

# Column names
print("Column names:", df.columns.tolist())
```

```python
# First few rows
print(df.head())

# Info summary
print("\nDataset Info:")
print(df.info())

# Descriptive statistics
print("\nDescriptive Statistics:")
print(df.describe())
```

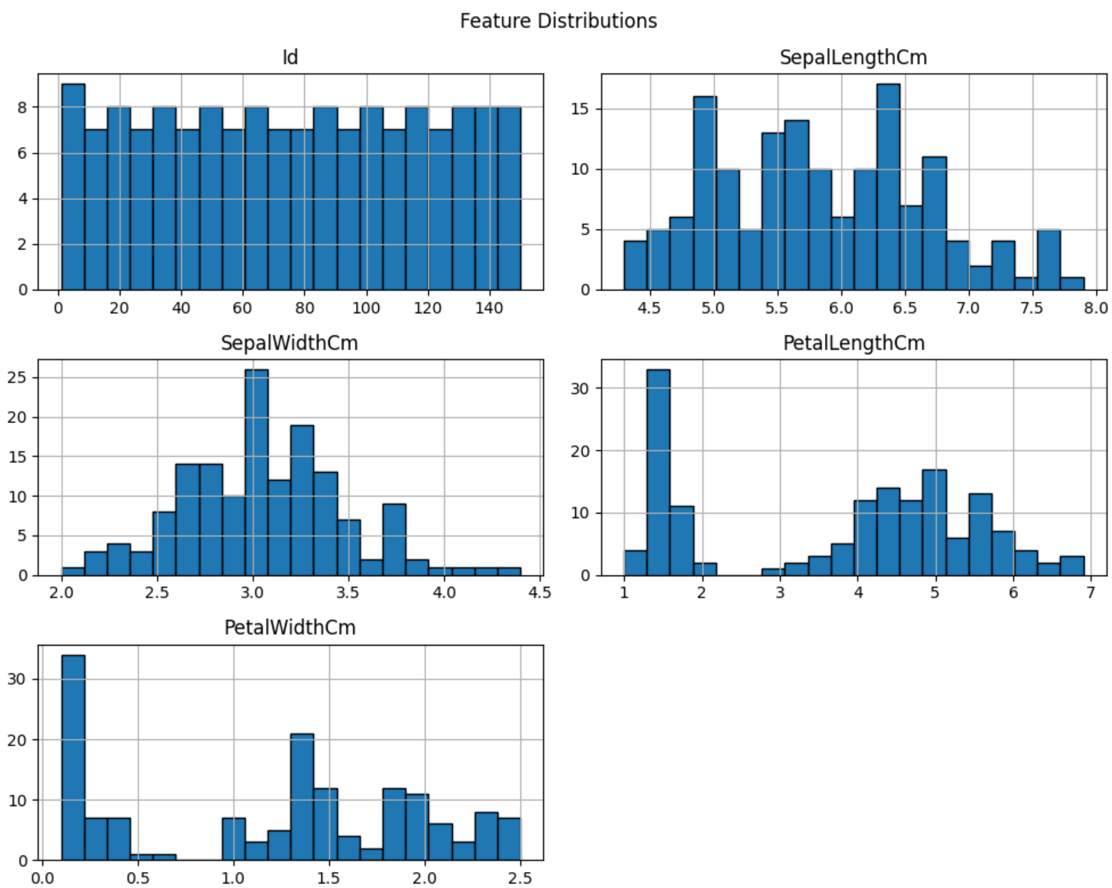
```
Shape of the dataset: (152, 6)
Column names: ['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm',
'PetalWidthCm', 'Species']
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0   1            5.1           3.5            NaN           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            NaN           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           NaN  Iris-setosa
4   5            5.0           NaN            1.4           0.2  Iris-setosa

Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 152 entries, 0 to 151
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             152 non-null    int64
 1   SepalLengthCm  151 non-null    float64
 2   SepalWidthCm   151 non-null    float64
 3   PetalLengthCm  151 non-null    float64
 4   PetalWidthCm   151 non-null    float64
 5   Species        152 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.3+ KB
None

Descriptive Statistics:
               Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count  152.000000     151.000000    151.000000     151.000000    151.000000
mean    75.414474       5.849007      3.055629       3.770861      1.206623
std     43.866813       0.823073      0.432302       1.764902      0.766870
min      1.000000       4.300000      2.000000       1.000000      0.100000
25%     37.750000       5.100000      2.800000       1.600000      0.300000
50%     75.500000       5.800000      3.000000       4.400000      1.300000
75%    113.250000       6.400000      3.300000       5.100000      1.800000
max    150.000000       7.900000      4.400000       6.900000      2.500000
```

#Visualize the Dataset

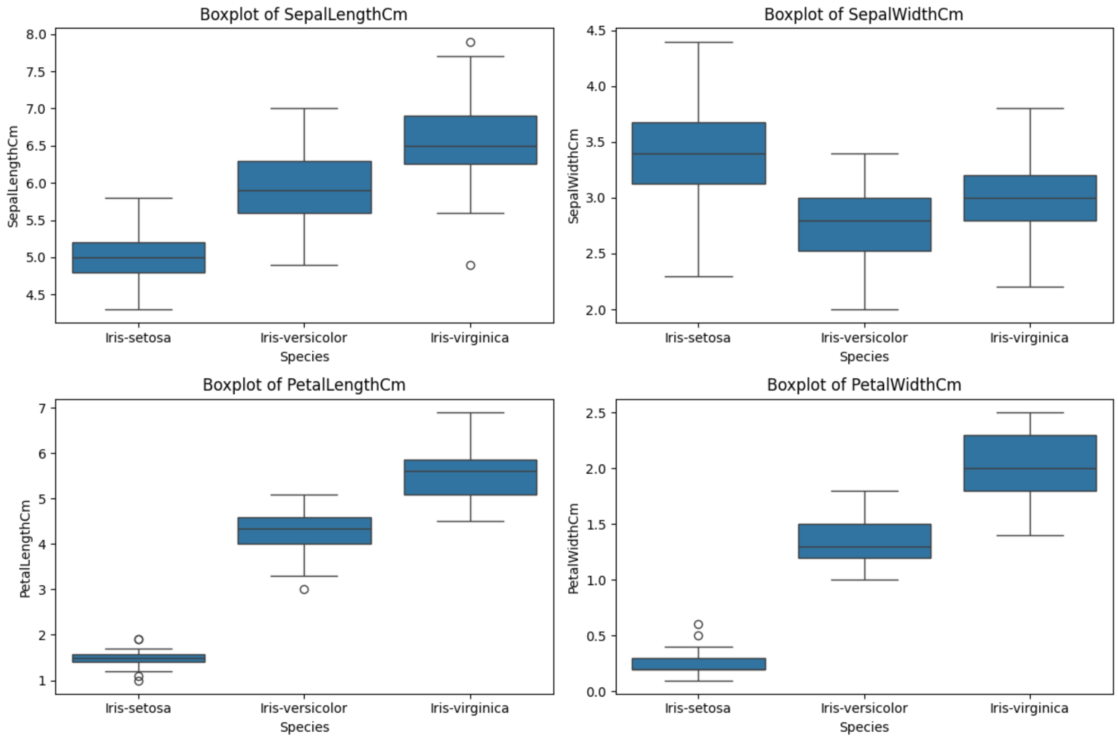
[32]:
```
#Scatter Plot -- Sepal Length vs Sepal Width

plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='SepalLengthCm', y='SepalWidthCm', hue='Species')
plt.title('Sepal Length vs Sepal Width')
plt.show()
```



[33]:
```
#Histograms -- Distribution of Each Feature
df.hist(figsize=(10, 8), bins=20, edgecolor='black')
plt.suptitle('Feature Distributions')
plt.tight_layout()
plt.show()
```



[34]:
```
#Boxplot -- To Identify Outlier
plt.figure(figsize=(12, 8))
features = ['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']

for i, column in enumerate(features):
    plt.subplot(2, 2, i + 1)
    sns.boxplot(x='Species', y=column, data=df)
    plt.title(f'Boxplot of {column}')
plt.tight_layout()
plt.show()
```



[34]:

```

```

5 Task 2: Predict Future Stock Prices (Short-Term)

[35]:

```
pip install yfinance
```

```
Requirement already satisfied: yfinance in /usr/local/lib/python3.11/dist-
packages (0.2.61)
Requirement already satisfied: pandas>=1.3.0 in /usr/local/lib/python3.11/dist-
packages (from yfinance) (2.2.2)
Requirement already satisfied: numpy>=1.16.5 in /usr/local/lib/python3.11/dist-
packages (from yfinance) (2.0.2)
Requirement already satisfied: requests>=2.31 in /usr/local/lib/python3.11/dist-
packages (from yfinance) (2.32.3)
Requirement already satisfied: multitasking>=0.0.7 in
/usr/local/lib/python3.11/dist-packages (from yfinance) (0.0.11)
Requirement already satisfied: platformdirs>=2.0.0 in
/usr/local/lib/python3.11/dist-packages (from yfinance) (4.3.8)
Requirement already satisfied: pytz>=2022.5 in /usr/local/lib/python3.11/dist-
packages (from yfinance) (2025.2)
Requirement already satisfied: frozendict>=2.3.4 in
/usr/local/lib/python3.11/dist-packages (from yfinance) (2.4.6)
Requirement already satisfied: peewee>=3.16.2 in /usr/local/lib/python3.11/dist-
```

```
packages (from yfinance) (3.18.1)
Requirement already satisfied: beautifulsoup4>=4.11.1 in
/usr/local/lib/python3.11/dist-packages (from yfinance) (4.13.4)
Requirement already satisfied: curl_cffi>=0.7 in /usr/local/lib/python3.11/dist-
packages (from yfinance) (0.11.1)
Requirement already satisfied: protobuf>=3.19.0 in
/usr/local/lib/python3.11/dist-packages (from yfinance) (5.29.5)
Requirement already satisfied: websockets>=13.0 in
/usr/local/lib/python3.11/dist-packages (from yfinance) (15.0.1)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.11/dist-
packages (from beautifulsoup4>=4.11.1->yfinance) (2.7)
Requirement already satisfied: typing-extensions>=4.0.0 in
/usr/local/lib/python3.11/dist-packages (from beautifulsoup4>=4.11.1->yfinance)
(4.13.2)
Requirement already satisfied: cffi>=1.12.0 in /usr/local/lib/python3.11/dist-
packages (from curl_cffi>=0.7->yfinance) (1.17.1)
Requirement already satisfied: certifi>=2024.2.2 in
/usr/local/lib/python3.11/dist-packages (from curl_cffi>=0.7->yfinance)
(2025.4.26)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas>=1.3.0->yfinance)
(2.9.0.post0)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-
packages (from pandas>=1.3.0->yfinance) (2025.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.31->yfinance) (3.4.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-
packages (from requests>=2.31->yfinance) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.31->yfinance) (2.4.0)
Requirement already satisfied: pycparser in /usr/local/lib/python3.11/dist-
packages (from cffi>=1.12.0->curl_cffi>=0.7->yfinance) (2.22)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-
packages (from python-dateutil>=2.8.2->pandas>=1.3.0->yfinance) (1.17.0)
```

#Import libraries and load data

[36]:
```
import yfinance as yf
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt

# Download historical data for Apple (AAPL)
ticker = 'AAPL'
```

```
df = yf.download(ticker, start='2020-01-01', end='2024-01-01')

# Display first rows
print(df.head())
```

```
[*********************100%***********************]  1 of 1 completed

Price           Close       High        Low       Open     Volume
Ticker           AAPL       AAPL       AAPL       AAPL       AAPL
Date
2020-01-02  72.620834  72.681281  71.373211  71.627084  135480400
2020-01-03  71.914795  72.676423  71.689935  71.847095  146322800
2020-01-06  72.487839  72.526526  70.783241  71.034702  118387200
2020-01-07  72.146950  72.753831  71.926922  72.497537  108872000
2020-01-08  73.307510  73.609745  71.849533  71.849533  132079200
```

6 Prepare features and target

[37]:
```
# Shift the Close column up by 1 to represent next day close price
df['Next_Close'] = df['Close'].shift(-1)

# Drop last row with NaN target
df = df[:-1]

# Features and target
features = ['Open', 'High', 'Low', 'Volume']
X = df[features]
y = df['Next_Close']
```

#Split data into train/test sets

[38]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, shuffle=False,
 ↪test_size=0.2)
```

7 Train the model

[39]:
```
#Linear Regression
model = LinearRegression()
model.fit(X_train, y_train)
```

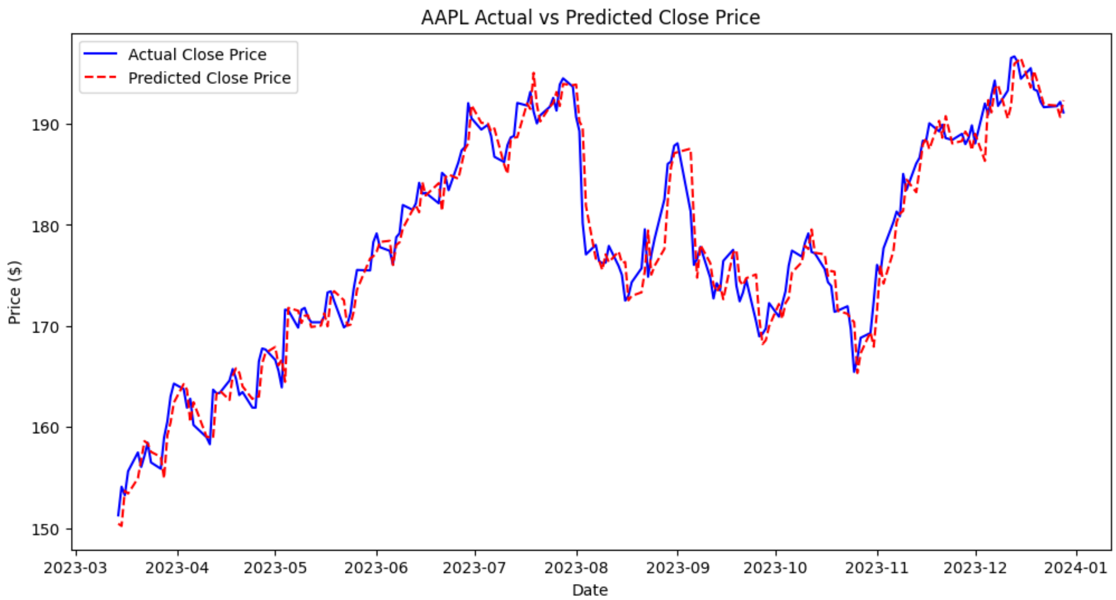
[39]: LinearRegression()

[40]:
```
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse:.4f}")
```

```
Mean Squared Error: 4.9761
```

#Plot actual vs predicted closing prices

[41]:
```
plt.figure(figsize=(12, 6))
plt.plot(y_test.index, y_test, label='Actual Close Price', color='blue')
plt.plot(y_test.index, y_pred, label='Predicted Close Price', color='red',␣
 ↪linestyle='--')
plt.title(f'{ticker} Actual vs Predicted Close Price')
plt.xlabel('Date')
plt.ylabel('Price ($)')
plt.legend()
plt.show()
```



[41]:
```

```

8 Task 3: Heart Disease Prediction

9 Import Libraries

[42]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

```python
from sklearn.metrics import accuracy_score, roc_auc_score, roc_curve,␣
↪confusion_matrix, ConfusionMatrixDisplay
```

10 Load Dataset

[43]:
```python
data = pd.read_csv('/content/archive.zip')
data.head()
```

[43]:
```
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   69    1   0       160   234    1        2      131      0      0.1      1
1   69    0   0       140   239    0        0      151      0      1.8      0
2   66    0   0       150   226    0        0      114      0      2.6      2
3   65    1   0       138   282    1        2      174      0      1.4      1
4   64    1   0       110   211    0        2      144      1      1.8      1

   ca  thal  condition
0   1     0          0
1   2     0          0
2   0     0          0
3   1     0          1
4   0     0          0
```

10.1 Check Missing Values

[44]:
```python
print("Missing values in each column:")
print(data.isnull().sum())
```

```
Missing values in each column:
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
condition    0
dtype: int64
```

11 Basic Info and Description

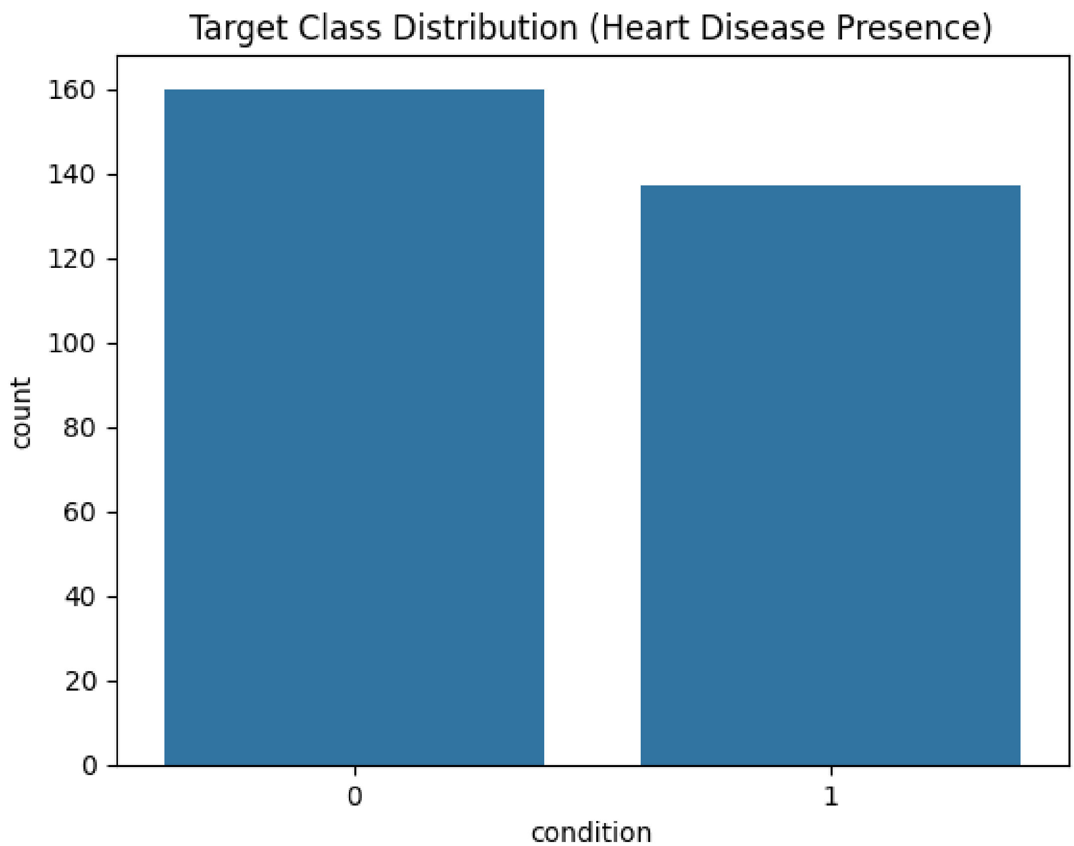
[45]:
```
print(data.info())
print(data.describe())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 297 entries, 0 to 296
Data columns (total 14 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   age        297 non-null    int64
 1   sex        297 non-null    int64
 2   cp         297 non-null    int64
 3   trestbps   297 non-null    int64
 4   chol       297 non-null    int64
 5   fbs        297 non-null    int64
 6   restecg    297 non-null    int64
 7   thalach    297 non-null    int64
 8   exang      297 non-null    int64
 9   oldpeak    297 non-null    float64
 10  slope      297 non-null    int64
 11  ca         297 non-null    int64
 12  thal       297 non-null    int64
 13  condition  297 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 32.6 KB
None
              age         sex          cp    trestbps        chol         fbs  \
count  297.000000  297.000000  297.000000  297.000000  297.000000  297.000000
mean    54.542088    0.676768    2.158249  131.693603  247.350168    0.144781
std      9.049736    0.468500    0.964859   17.762806   51.997583    0.352474
min     29.000000    0.000000    0.000000   94.000000  126.000000    0.000000
25%     48.000000    0.000000    2.000000  120.000000  211.000000    0.000000
50%     56.000000    1.000000    2.000000  130.000000  243.000000    0.000000
75%     61.000000    1.000000    3.000000  140.000000  276.000000    0.000000
max     77.000000    1.000000    3.000000  200.000000  564.000000    1.000000

          restecg     thalach       exang     oldpeak       slope          ca  \
count  297.000000  297.000000  297.000000  297.000000  297.000000  297.000000
mean     0.996633  149.599327    0.326599    1.055556    0.602694    0.676768
std      0.994914   22.941562    0.469761    1.166123    0.618187    0.938965
min      0.000000   71.000000    0.000000    0.000000    0.000000    0.000000
25%      0.000000  133.000000    0.000000    0.000000    0.000000    0.000000
50%      1.000000  153.000000    0.000000    0.800000    1.000000    0.000000
75%      2.000000  166.000000    1.000000    1.600000    1.000000    1.000000
max      2.000000  202.000000    1.000000    6.200000    2.000000    3.000000
```

```
             thal   condition
count  297.000000  297.000000
mean     0.835017    0.461279
std      0.956690    0.499340
min      0.000000    0.000000
25%      0.000000    0.000000
50%      0.000000    0.000000
75%      2.000000    1.000000
max      2.000000    1.000000
```

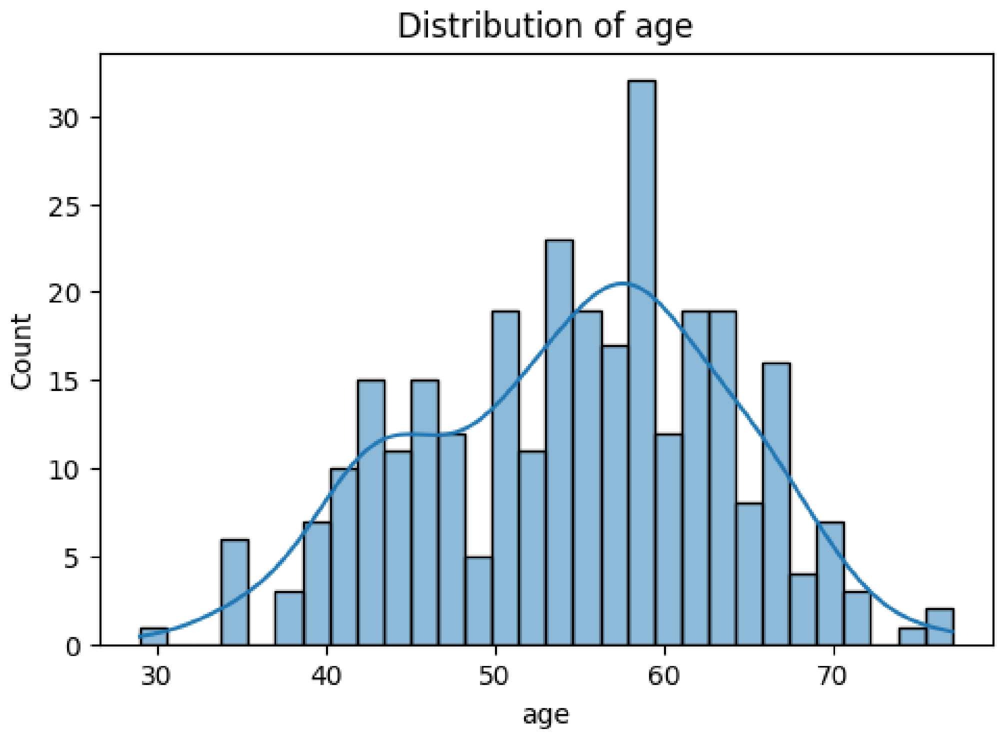
12 Visualize Target Distribution

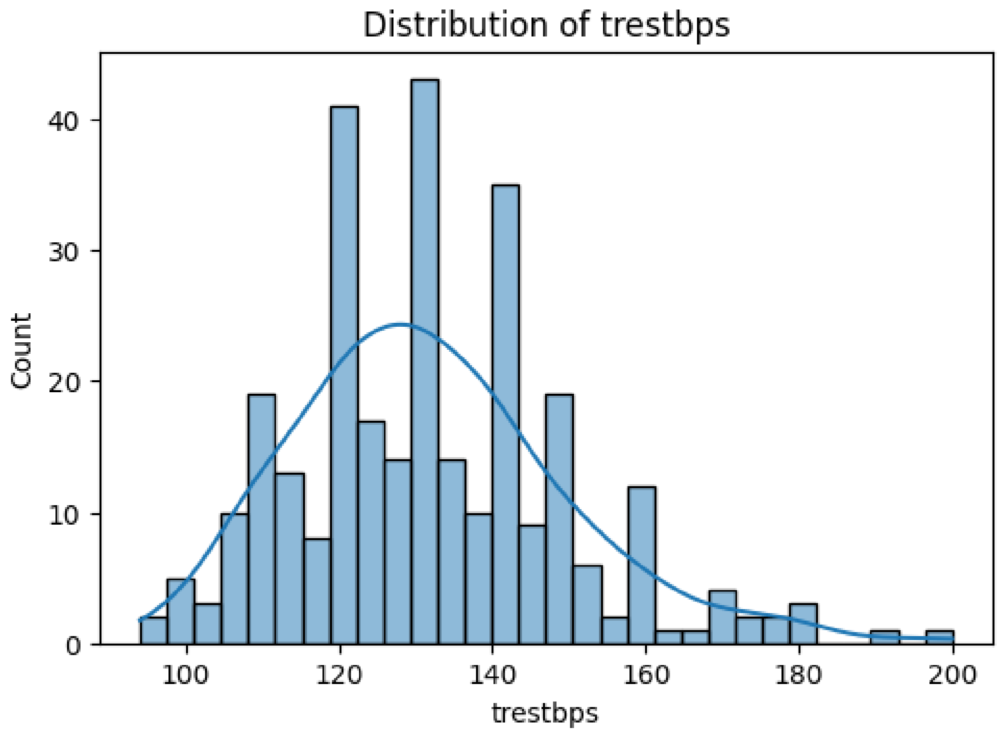
[48]:
```python
sns.countplot(x='condition', data=data)
plt.title('Target Class Distribution (Heart Disease Presence)')
plt.show()
```

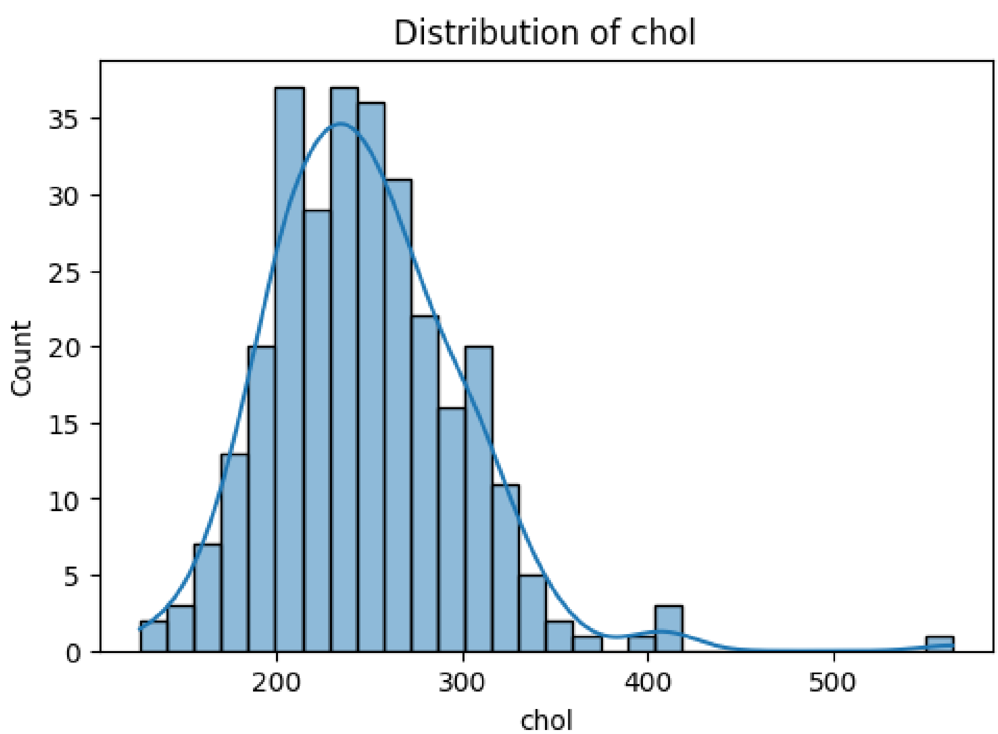


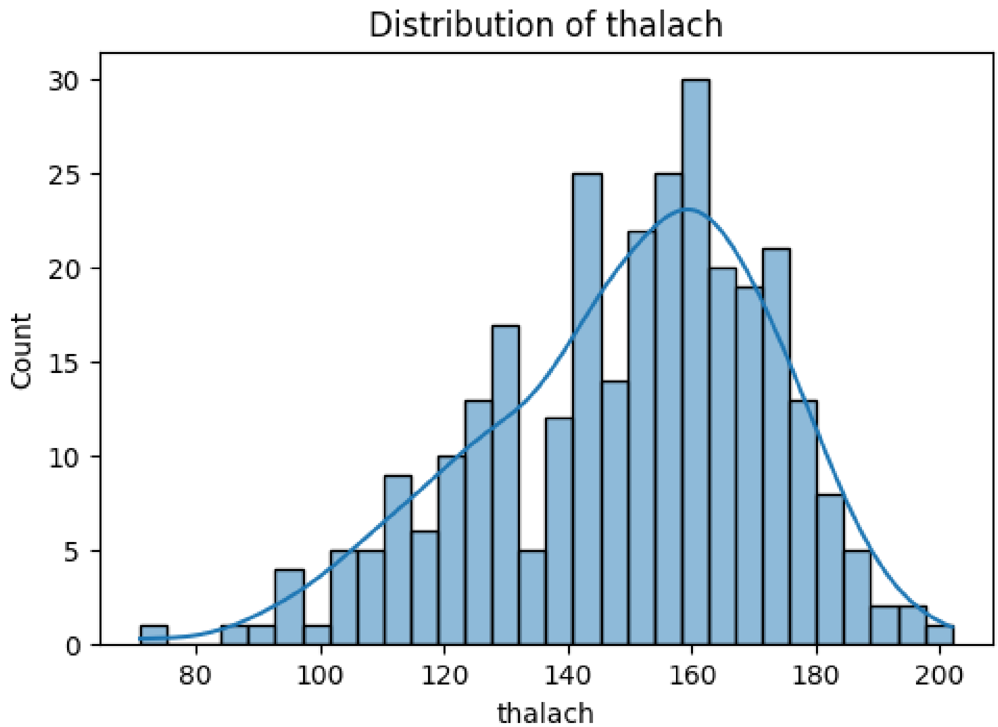
13 Distribution of Numerical Features

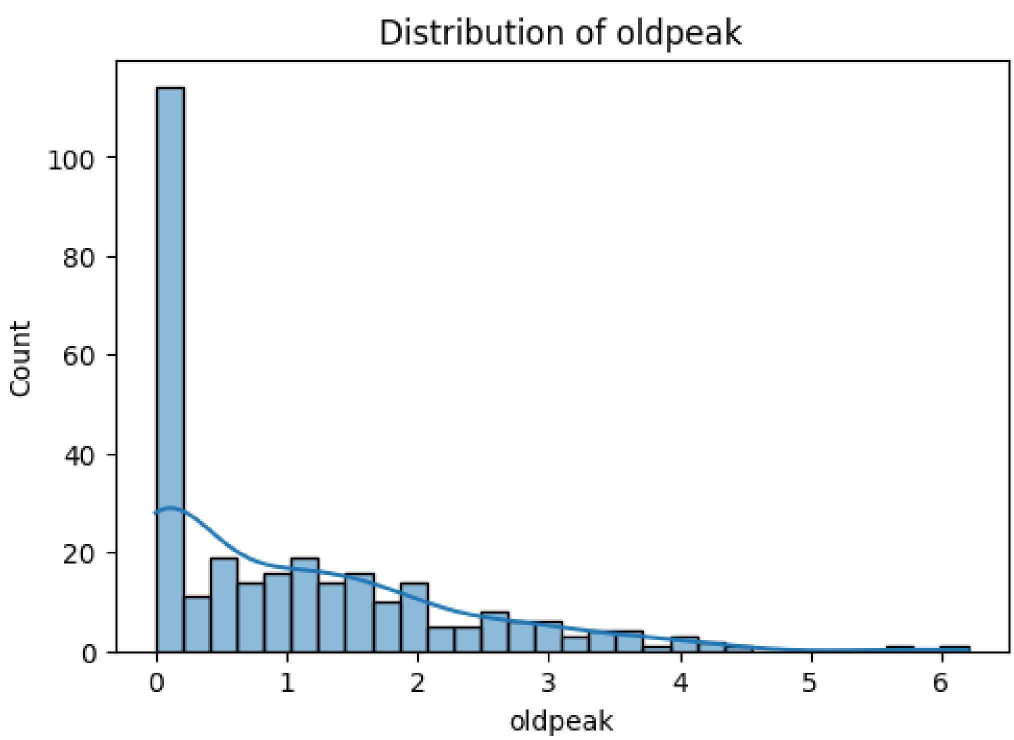
[59]:
```python
numerical_cols = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']

for col in numerical_cols:
    plt.figure(figsize=(6, 4))
    sns.histplot(data[col], kde=True, bins=30)
    plt.title(f'Distribution of {col}')
    plt.show()
```




Distribution of trestbps
Count
0
10
20
30
40
100
120
140
160
180
200
trestbps

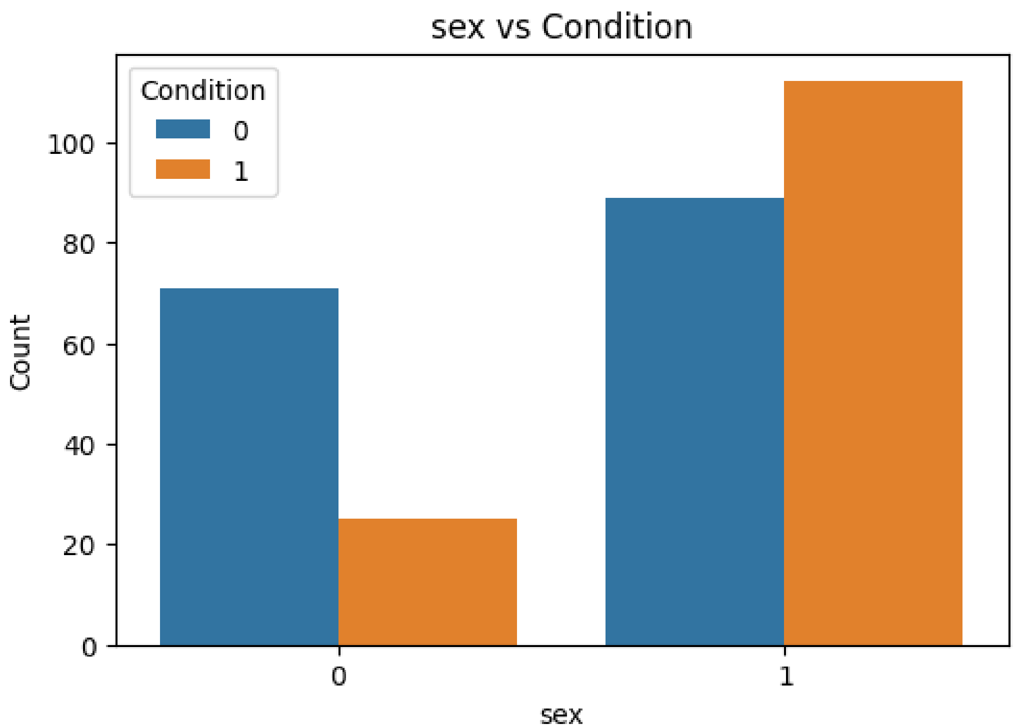

Distribution of chol
Count
0
5
10
15
20
25
30
35
200
300
400
500
chol

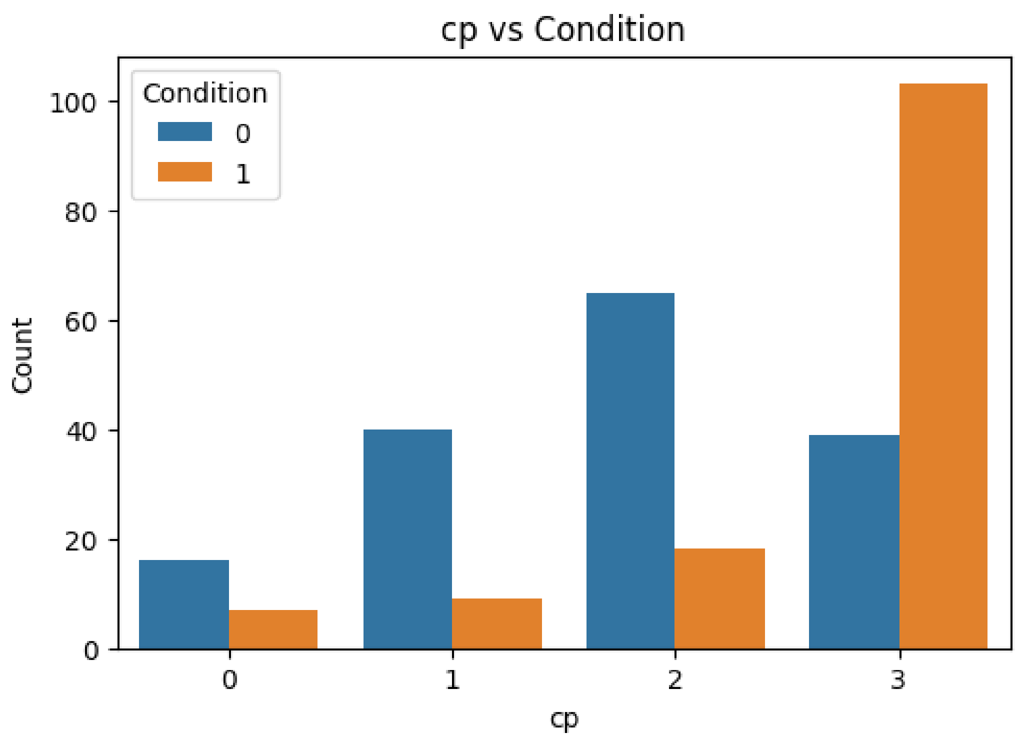

Distribution of thalach
Count
thalach
0
5
10
15
20
25
30
80
100
120
140
160
180
200

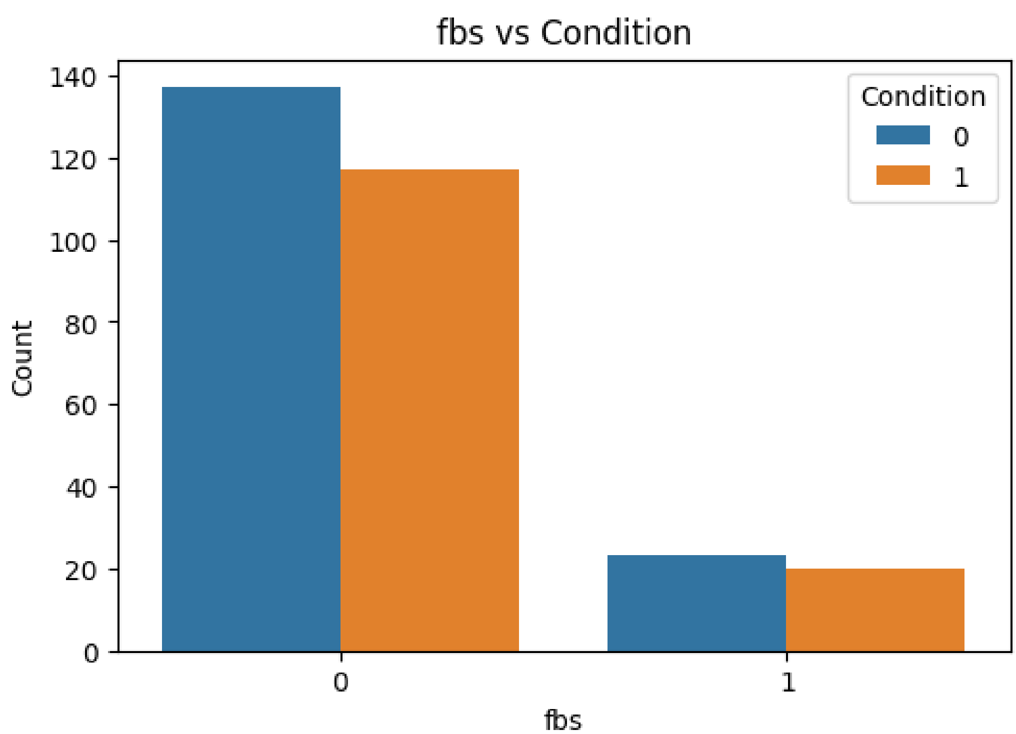

Distribution of oldpeak
Count
oldpeak
0
20
40
60
80
100
0
1
2
3
4
5
6

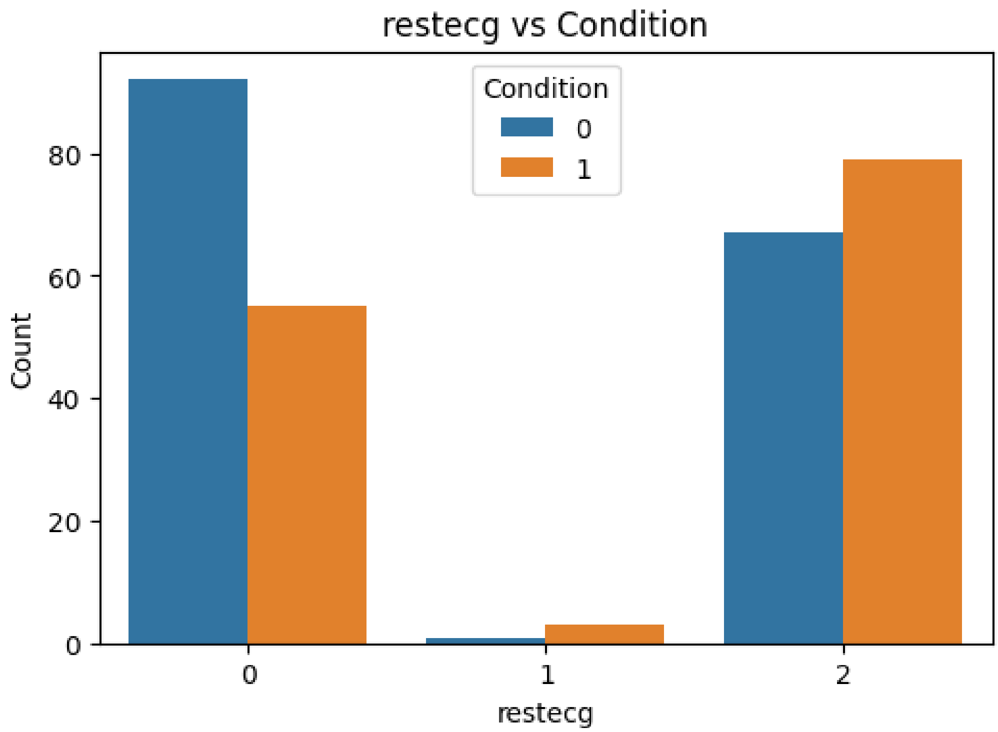
14 Categorical Features vs Target

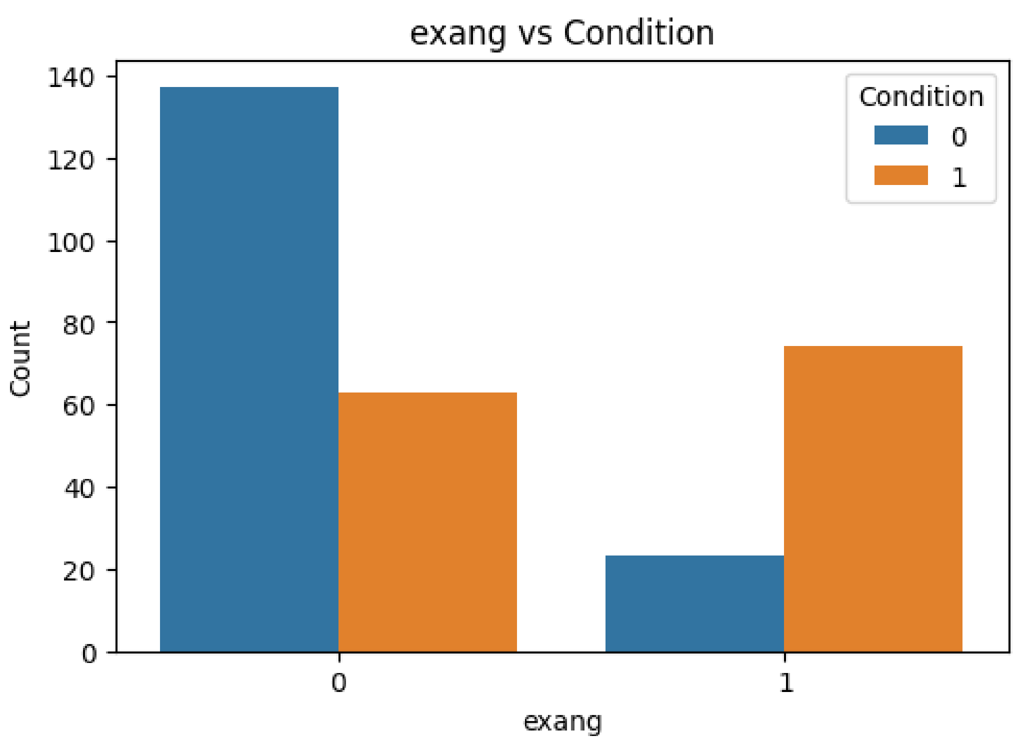
[60]:
```
categorical_cols = ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca',␣
↪'thal']

for col in categorical_cols:
    plt.figure(figsize=(6, 4))
    sns.countplot(x=col, hue='condition', data=data)
    plt.title(f'{col} vs Condition')
    plt.xlabel(col)
    plt.ylabel('Count')
    plt.legend(title='Condition')
    plt.show()
```

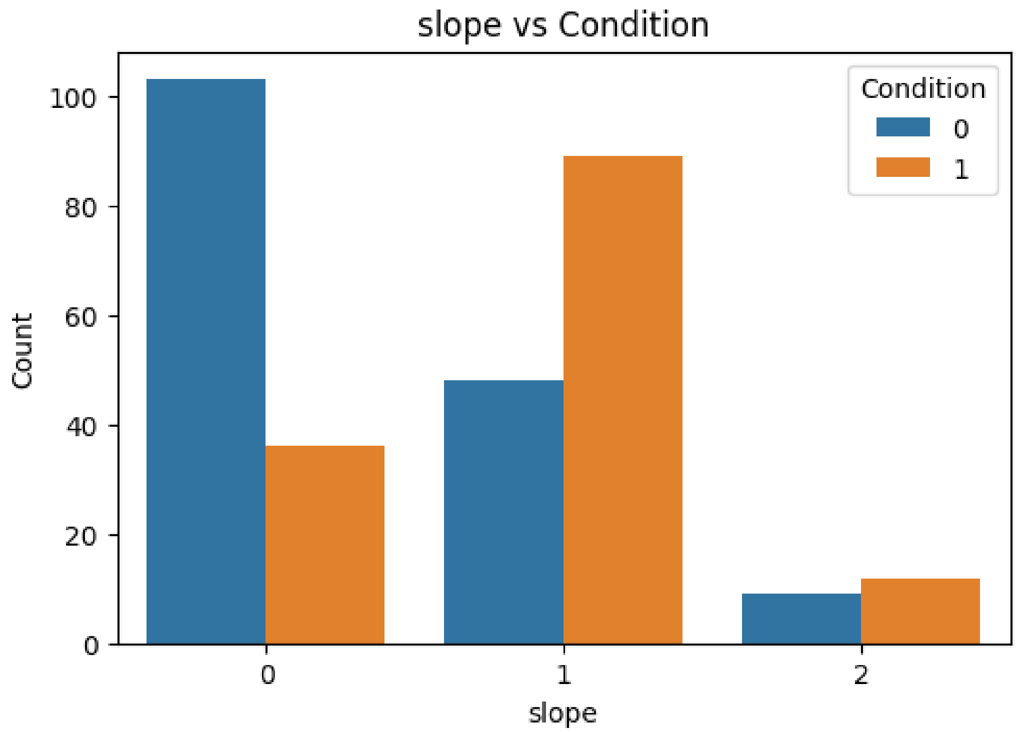


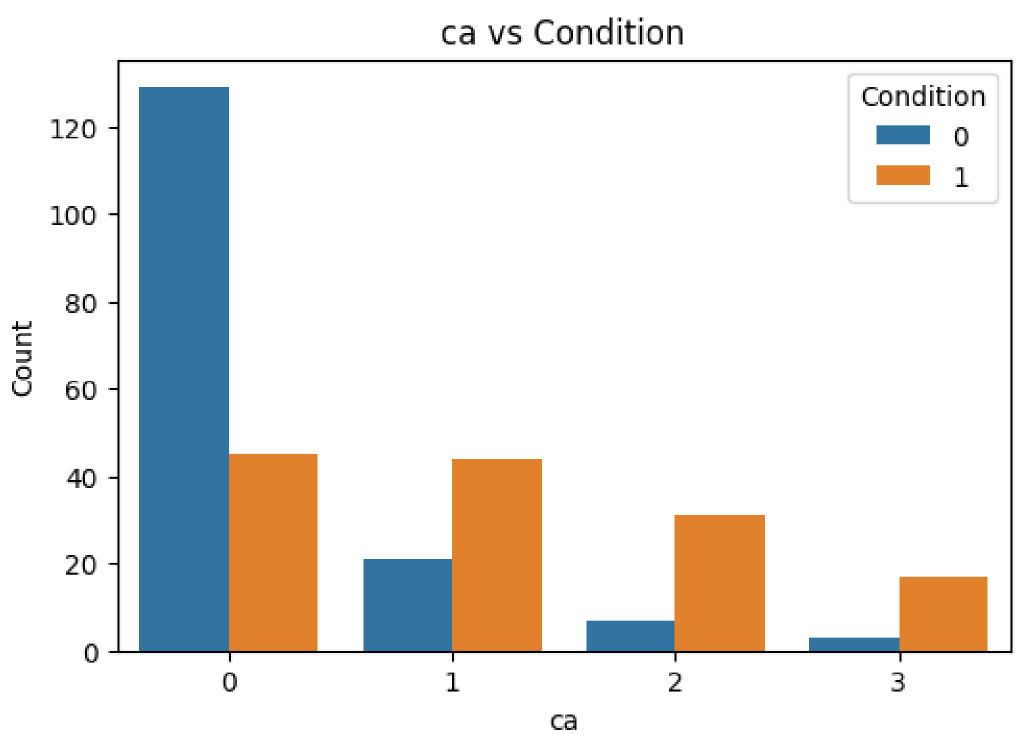

cp vs Condition
Condition
0
1
Count
100
80
60
40
20
0
0
1
2
3
cp

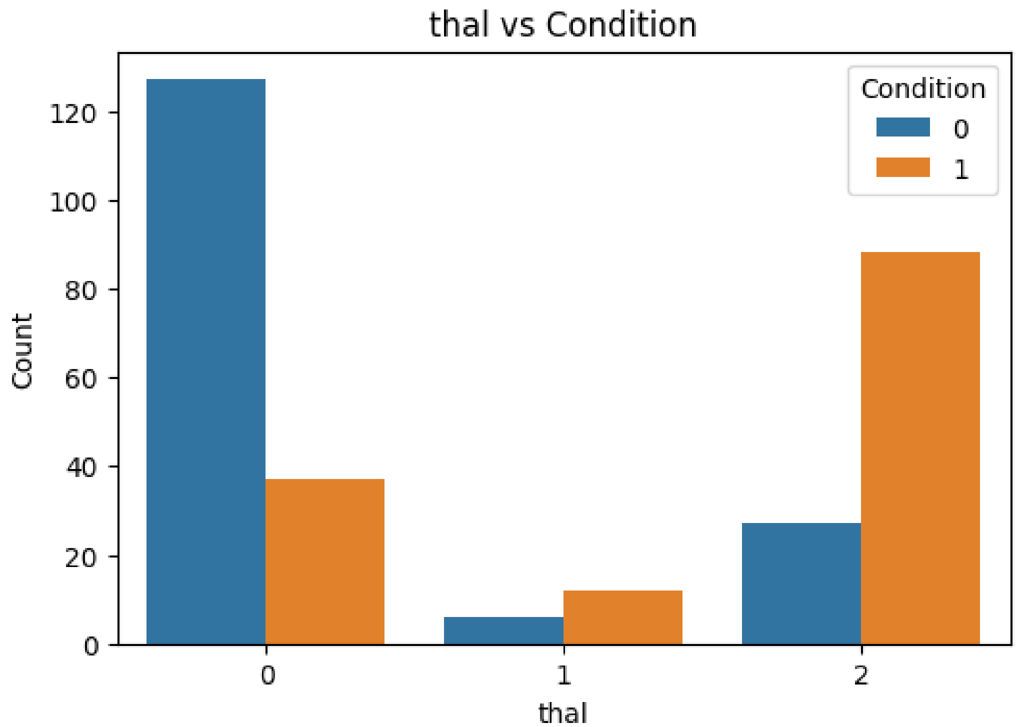

fbs vs Condition
Condition
0
1
Count
140
120
100
80
60
40
20
0
0
1
fbs


restecg vs Condition
Condition
0
1
Count
0
20
40
60
80
0
1
2
restecg


exang vs Condition
Condition
0
1
Count
0
20
40
60
80
100
120
140
0
1
exang

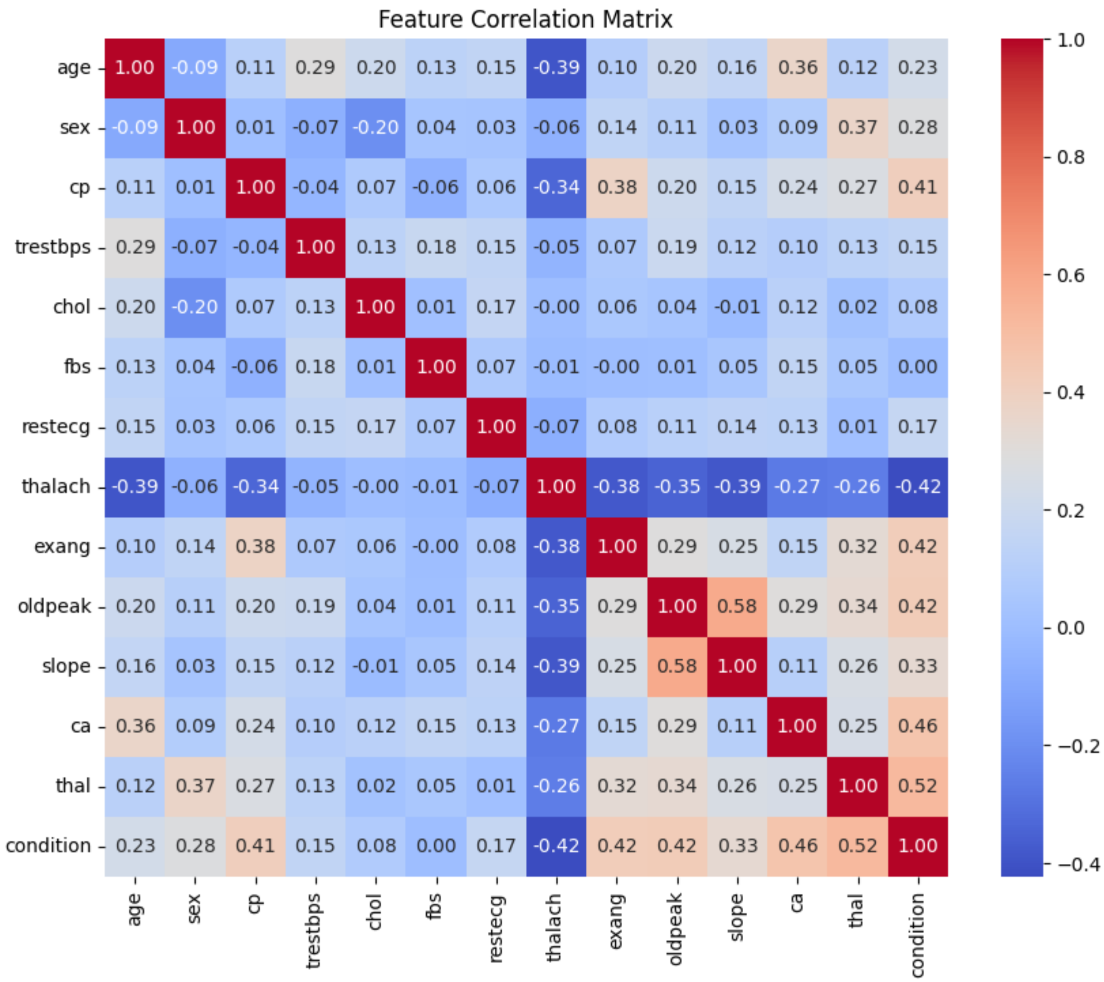

slope vs Condition
Condition
0
1
Count
0
20
40
60
80
100
0
1
2
slope


ca vs Condition
Condition
0
1
Count
0
20
40
60
80
100
120
0
1
2
3
ca



15 Correlation Heatmap

[49]:
```
plt.figure(figsize=(10,8))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Feature Correlation Matrix')
plt.show()
```



16 Prepare Data for Modeling

[52]:
```
X = data.drop('condition', axis=1)
y = data['condition']

# Train-test split (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=42, stratify=y)
```

17 Train Logistic Regression Model

[53]:
```
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

[53]: LogisticRegression(max_iter=1000)

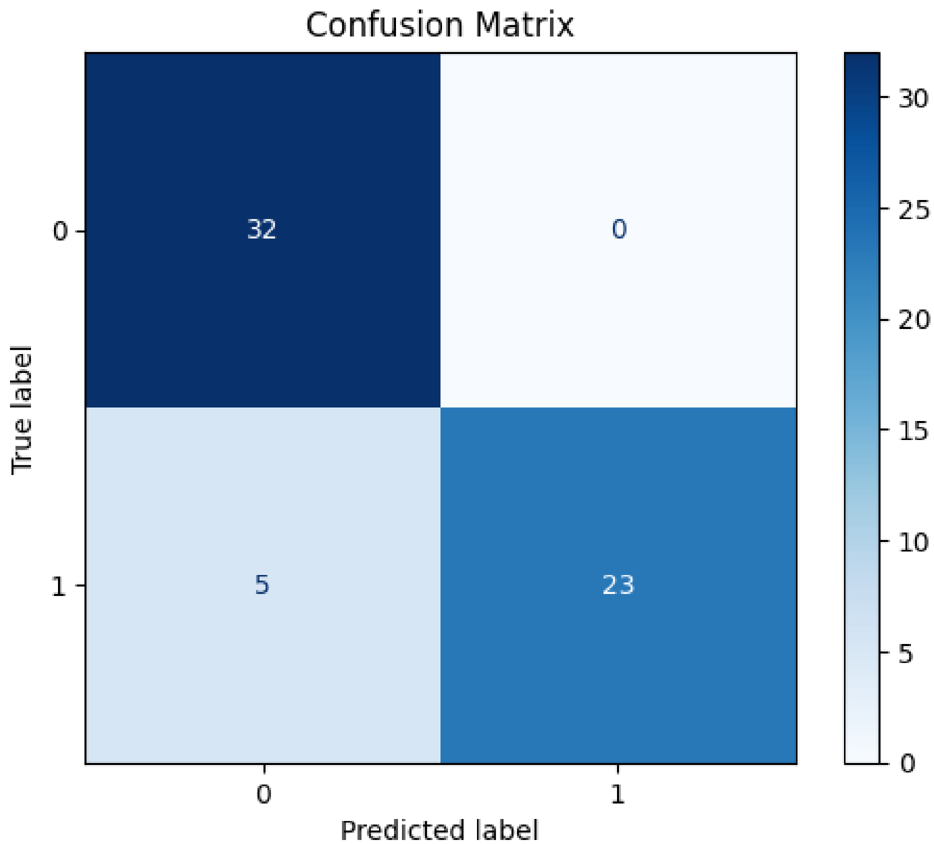
18 Make Predictions and Evaluate

[54]:
```
y_pred = model.predict(X_test)
y_prob = model.predict_proba(X_test)[:, 1]

accuracy = accuracy_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_prob)

print(f'Accuracy: {accuracy:.4f}')
print(f'ROC AUC: {roc_auc:.4f}')
```

```
Accuracy: 0.9167
ROC AUC: 0.9509
```

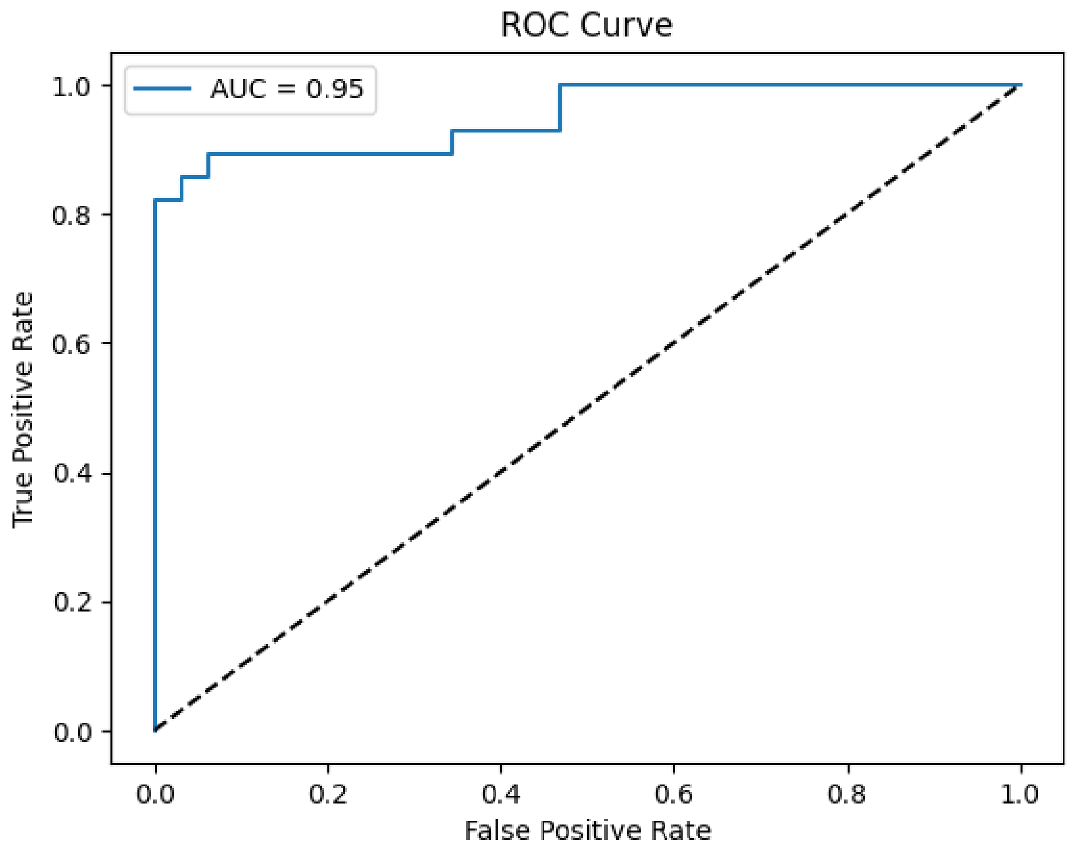
19 Plot Confusion Matrix

[55]:
```
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot(cmap='Blues')
plt.title('Confusion Matrix')
plt.show()
```



20 Plot ROC Curve

[56]:
```
fpr, tpr, thresholds = roc_curve(y_test, y_prob)

plt.plot(fpr, tpr, label=f'AUC = {roc_auc:.2f}')
plt.plot([0,1], [0,1], 'k--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()
```



21 Feature Importance (Logistic Regression Coefficients)

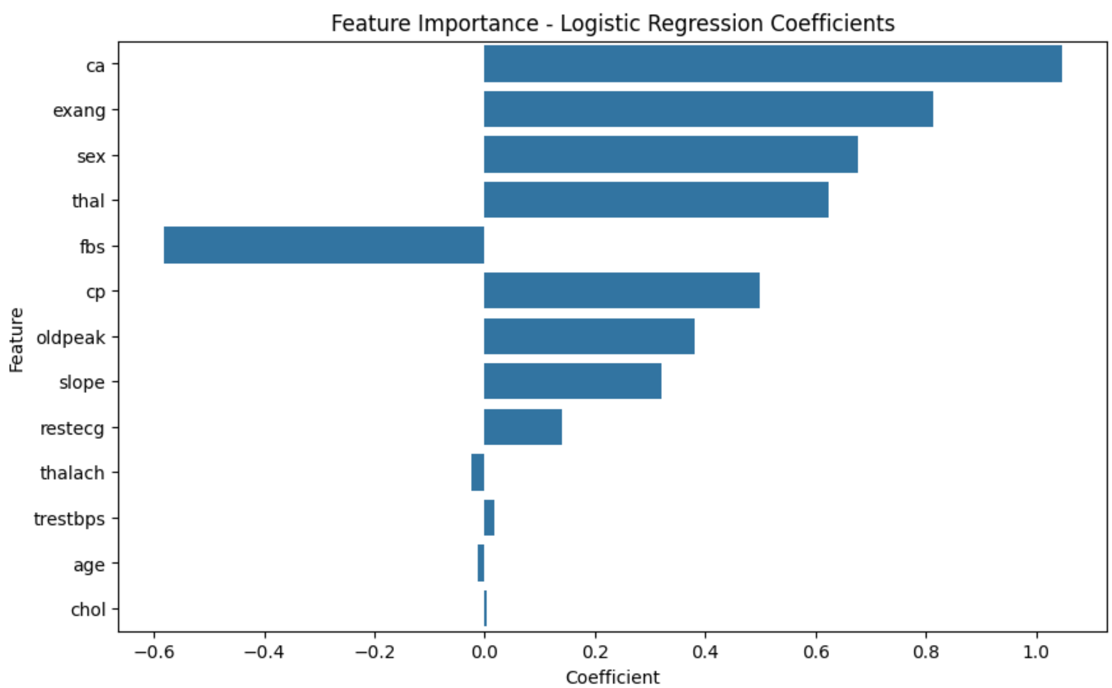
[57]:
```python
features = X.columns
coefficients = model.coef_[0]

importance_df = pd.DataFrame({'Feature': features, 'Coefficient': coefficients})
importance_df['AbsCoefficient'] = importance_df['Coefficient'].abs()
importance_df = importance_df.sort_values(by='AbsCoefficient', ascending=False)

print("Important Features (based on logistic regression coefficients):")
print(importance_df[['Feature', 'Coefficient']])
```

```
Important Features (based on logistic regression coefficients):
     Feature  Coefficient
11        ca     1.046757
8      exang     0.812330
1        sex     0.675442
12      thal     0.623016
5        fbs    -0.582774
```

```
2          cp     0.498075
9     oldpeak     0.379411
10      slope     0.320660
6     restecg     0.139551
7     thalach    -0.023170
3    trestbps     0.017056
0         age    -0.013472
4        chol     0.002924
```

22 Visualize Feature Importance

[58]:
```python
plt.figure(figsize=(10,6))
sns.barplot(x='Coefficient', y='Feature', data=importance_df)
plt.title('Feature Importance - Logistic Regression Coefficients')
plt.show()
```



[ ]:
```python

```

[62]:
```python
import getpass
username = "zoya4477"
token = getpass.getpass("Enter your GitHub token: ")
```

```
Enter your GitHub token: ··········
```

[63]:

```
!git add .
!git commit -m "AI-ML Tasks"
```

```
fatal: not a git repository (or any of the parent directories): .git
fatal: not a git repository (or any of the parent directories): .git
```

[64]:

```
!git pull https://github.com/zoya4477/AI-Ml.git
!git push https://{username}:{token}@github.com/zoya4477/AI-Ml.git
```

```
fatal: not a git repository (or any of the parent directories): .git
fatal: not a git repository (or any of the parent directories): .git
```

[ ]:

```
```