## Double-click (or enter) to edit

```
<> ←>
                                     Name: - Ravi Kant Tiwari
                                               Name:- Ravi Kant Tiwari Sec:- C University Roll
Sec:- C
                                               no:- 202410116100164
University Roll no: - 202410116100164
# STEP 1: Install and Import Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model selection import train test split
from sklearn.linear model import LinearRegression
from sklearn.metrics import mean squared error, r2 score
import warnings
warnings.filterwarnings("ignore")
# STEP 2: Load Dataset
# Load the dataset from the original source as suggested in the error message
data url = "http://lib.stat.cmu.edu/datasets/boston"
raw_df = pd.read_csv(data_url, sep="\s+", skiprows=22, header=None)
data = np.hstack([raw_df.values[::2, :], raw_df.values[1::2, :2]])
target = raw df.values[1::2, 2]
# Create a pandas DataFrame
df = pd.DataFrame(data, columns=['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', '
df['PRICE'] = target # Add the target variable
df.head()
# STEP 3: Explore Dataset
print(df.head())
print("\nShape:", df.shape)
print("\nMissing values:\n", df.isnull().sum())
# STEP 4: Correlation Heatmap
plt.figure(figsize=(12,10))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.title("Feature Correlation")
plt.show()
```

```
# STEP 5: Prepare Data
X = df.drop('PRICE', axis=1)
y = df['PRICE']
# STEP 6: Train/Test Split
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
# STEP 7: Train the Model
model = LinearRegression()
model.fit(X_train, y_train)
# STEP 8: Predictions
y_pred = model.predict(X_test)
# STEP 9: Evaluation
print("Mean Squared Error:", mean squared error(y test, y pred))
print("R^2 Score:", r2_score(y_test, y_pred))
# STEP 10: Plot Actual vs Predicted
plt.figure(figsize=(8,6))
plt.scatter(y test, y pred, c='crimson')
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual vs Predicted")
plt.grid(True)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='blue')
plt.show()
```



	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	\
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	

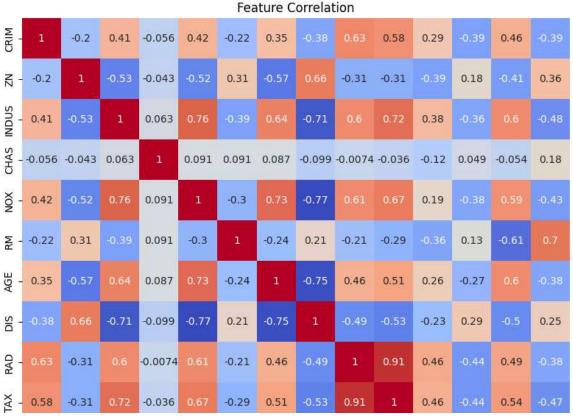
	PTRATIO	В	LSTAT	PRICE
0	15.3	396.90	4.98	24.0
1	17.8	396.90	9.14	21.6
2	17.8	392.83	4.03	34.7
3	18.7	394.63	2.94	33.4
4	18.7	396.90	5.33	36.2

Shape: (506, 14)

## Missing values:

CRIM	6
ZN	0
INDUS	0
CHAS	0
NOX	0
RM	0
AGE	0
DIS	0
RAD	0
TAX	0
PTRATIO	0
В	0
LSTAT	0
PRICE	0
dtype: inte	54





1.0

- 0.8

- 0.6

- 0.4

- 0.2

- 0.0

-0.2