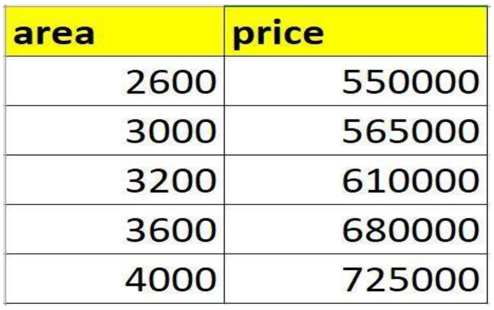
LAB #06

SUPERVISEDLEARNING(LINEARREGRESSION)

OBJECTIVES:

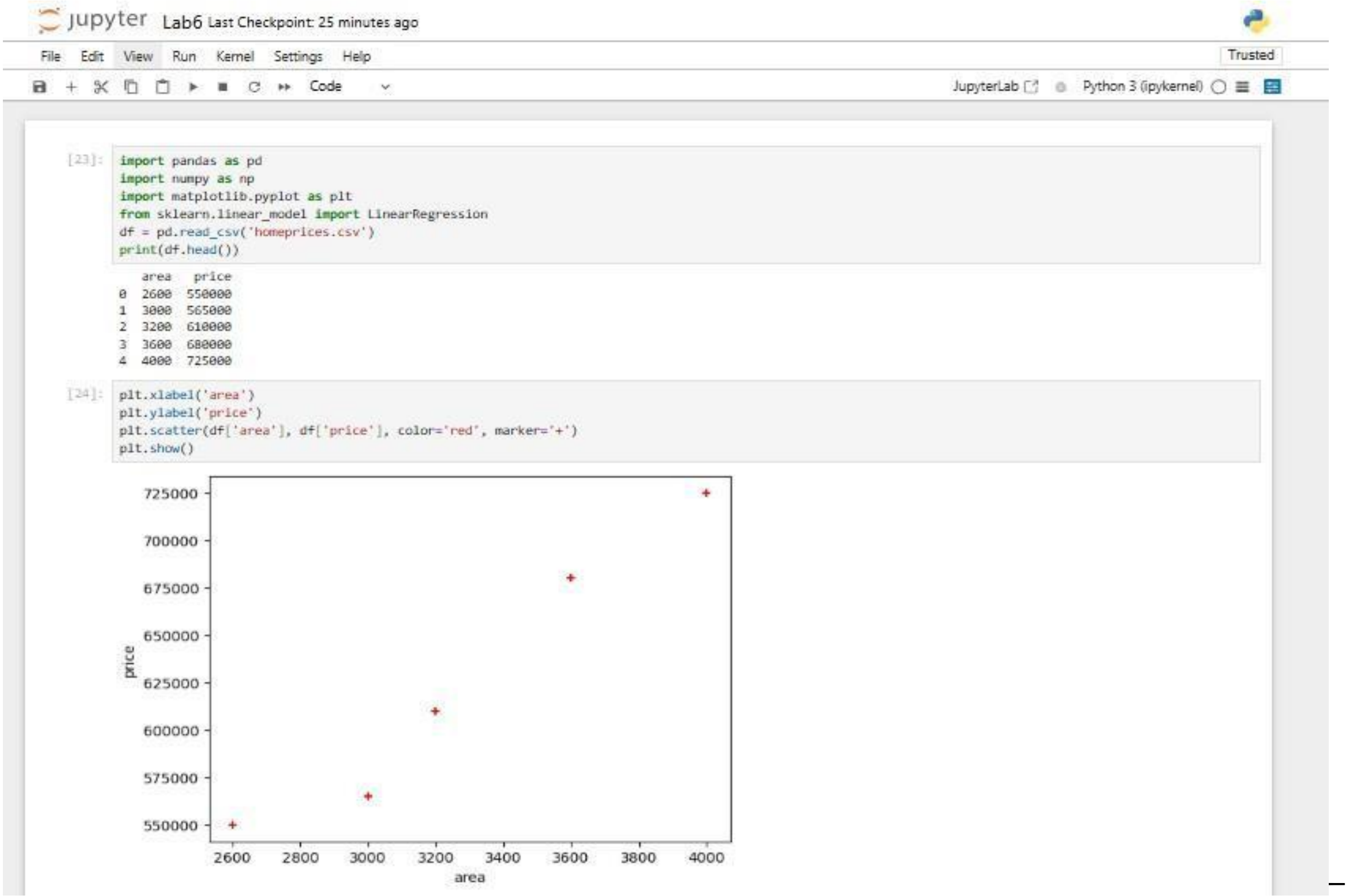
Implementing supervised learning, linear regression algorithm for training, testing and classification.

<u>Lab Task</u>



area	price
2600	550000
3000	565000
3200	610000
3600	680000
4000	725000

1. Implement linear regression algorithm on above dataset predict price of home with areas in the dataset by using (homeprices.csv).

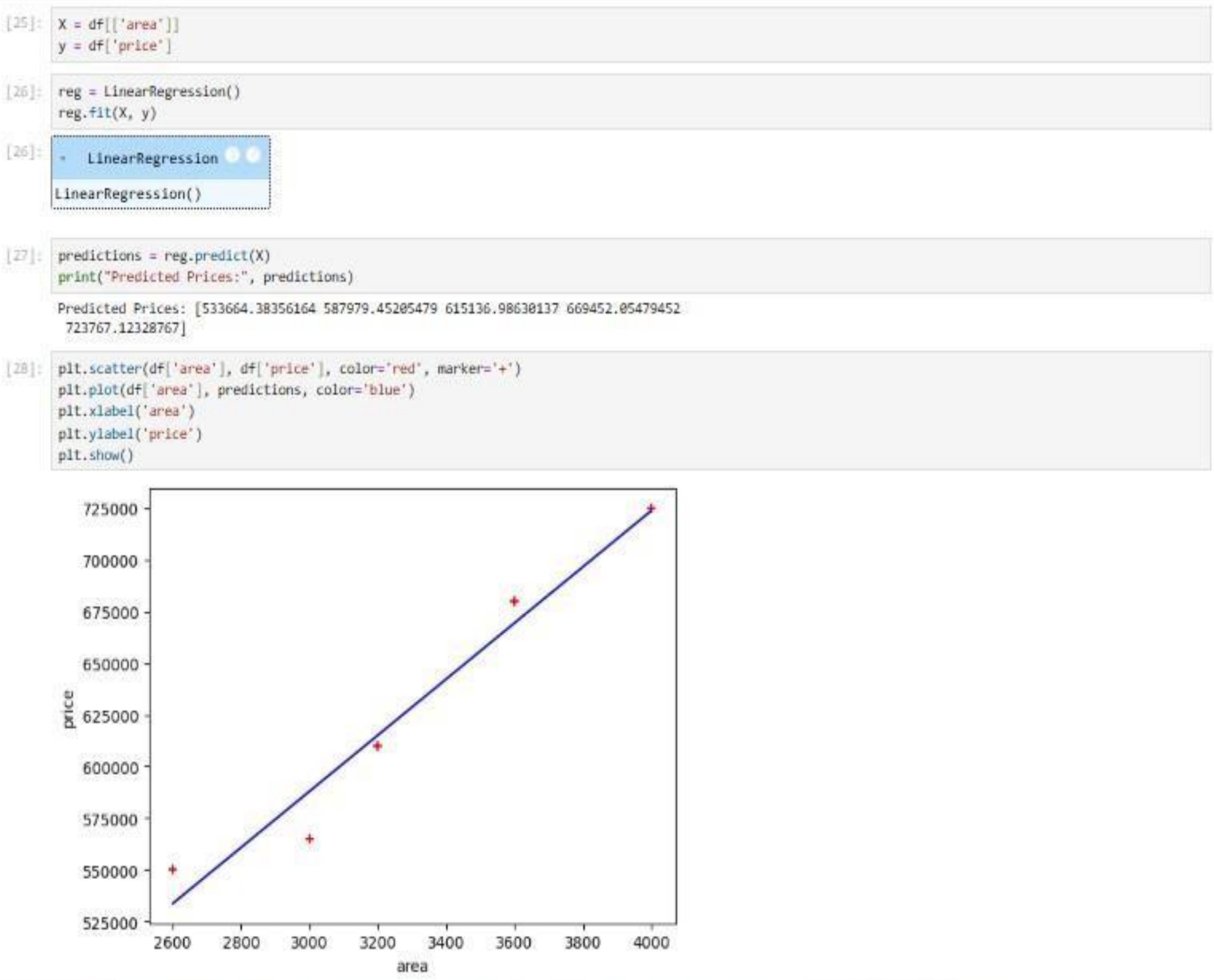


Jupyter Lab6 Last Checkpoint: 25 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

[23]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
df = pd.read_csv('homeprices.csv')
print(df.head())
```

```
   area   price
0  2600  550000
1  3000  565000
2  3200  610000
3  3600  680000
4  4000  725000
```

[24]:
```
plt.xlabel('area')
plt.ylabel('price')
plt.scatter(df['area'], df['price'], color='red', marker='+')
plt.show()
```



```
[25]: X = df[['area']]
      y = df['price']
```

```
[26]: reg = LinearRegression()
      reg.fit(X, y)
```

```
[26]: ▸ LinearRegression
      LinearRegression()
```

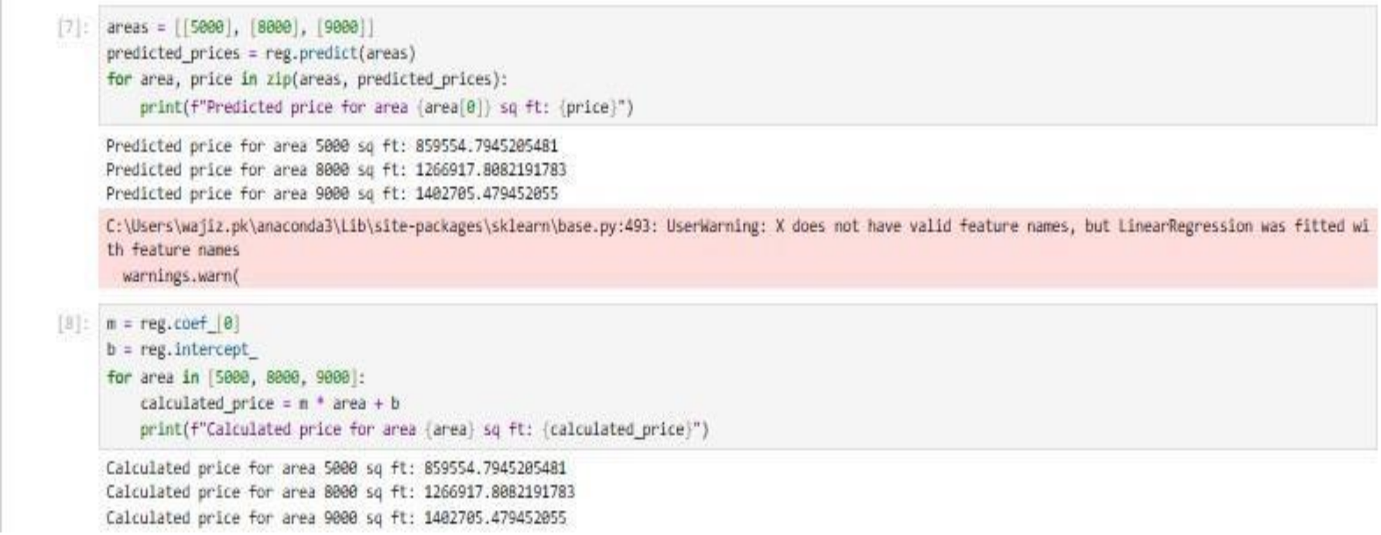
```
[27]: predictions = reg.predict(X)
      print("Predicted Prices:", predictions)
```

```
Predicted Prices: [533664.38356164 587979.45205479 615136.98630137 669452.05479452
 723767.12328767]
```

```
[28]: plt.scatter(df['area'], df['price'], color='red', marker='+')
      plt.plot(df['area'], predictions, color='blue')
      plt.xlabel('area')
      plt.ylabel('price')
      plt.show()
```

2. Implement linear regression using table1 in such a way that the:

Predict price of a home with area = 5000 Sqr. Ft.
Predict price of a home with area = 8000 Sqr. Ft.
Predict price of a home with area = 9000 Sqr. Ft.



```
[7]: areas = [[5000], [8000], [9000]]
     predicted_prices = reg.predict(areas)
     for area, price in zip(areas, predicted_prices):
         print(f"Predicted price for area {area[0]} sq ft: {price}")
```

```
Predicted price for area 5000 sq ft: 859554.7945205481
Predicted price for area 8000 sq ft: 1266917.8082191783
Predicted price for area 9000 sq ft: 1402705.479452055
C:\Users\wajiz.pk\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but LinearRegression was fitted wi
th feature names
  warnings.warn(
```

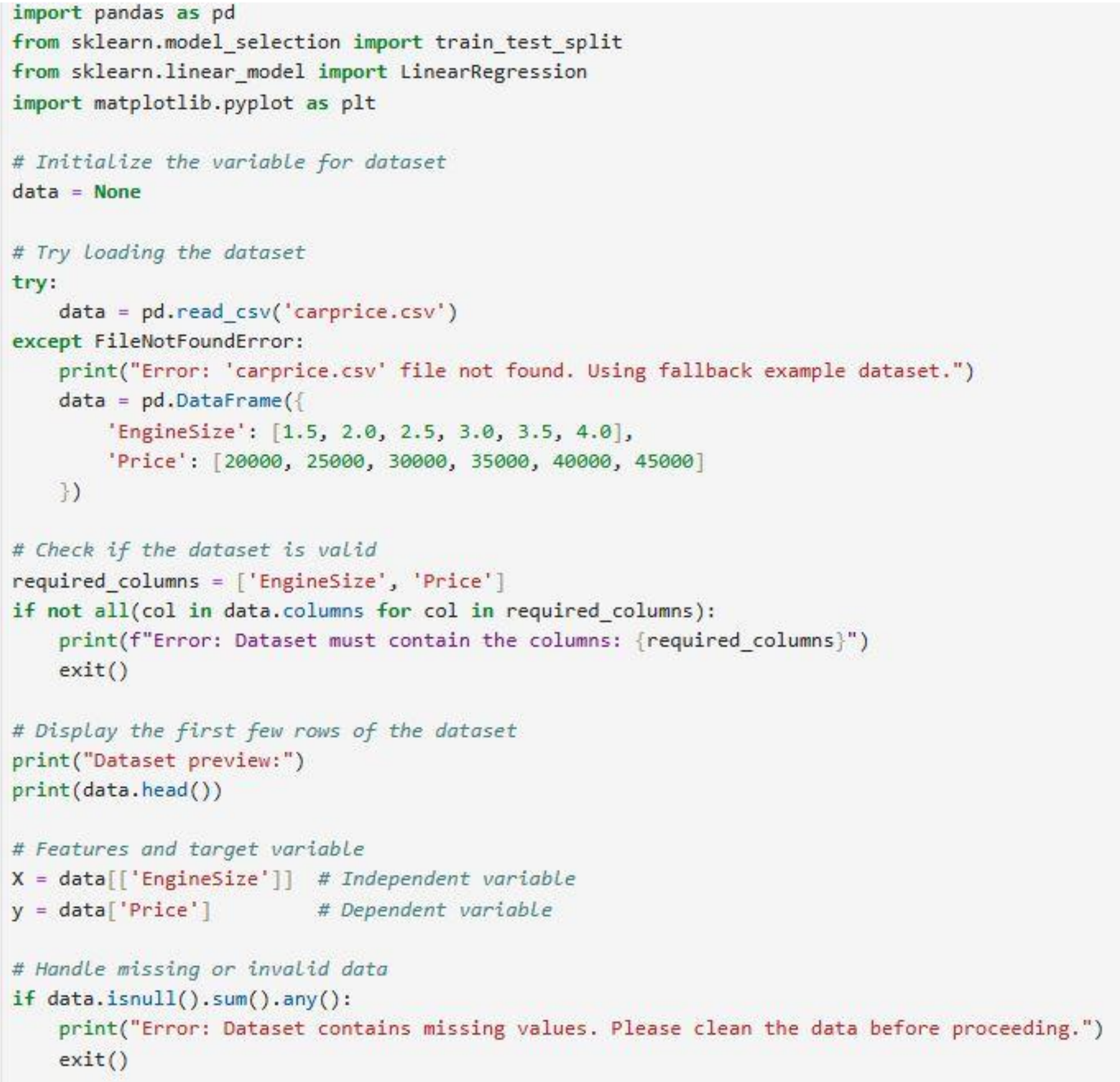
```
[8]: m = reg.coef_[0]
     b = reg.intercept_
     for area in [5000, 8000, 9000]:
         calculated_price = m * area + b
         print(f"Calculated price for area {area} sq ft: {calculated_price}")
```

```
Calculated price for area 5000 sq ft: 859554.7945205481
Calculated price for area 8000 sq ft: 1266917.8082191783
Calculated price for area 9000 sq ft: 1402705.479452055
```

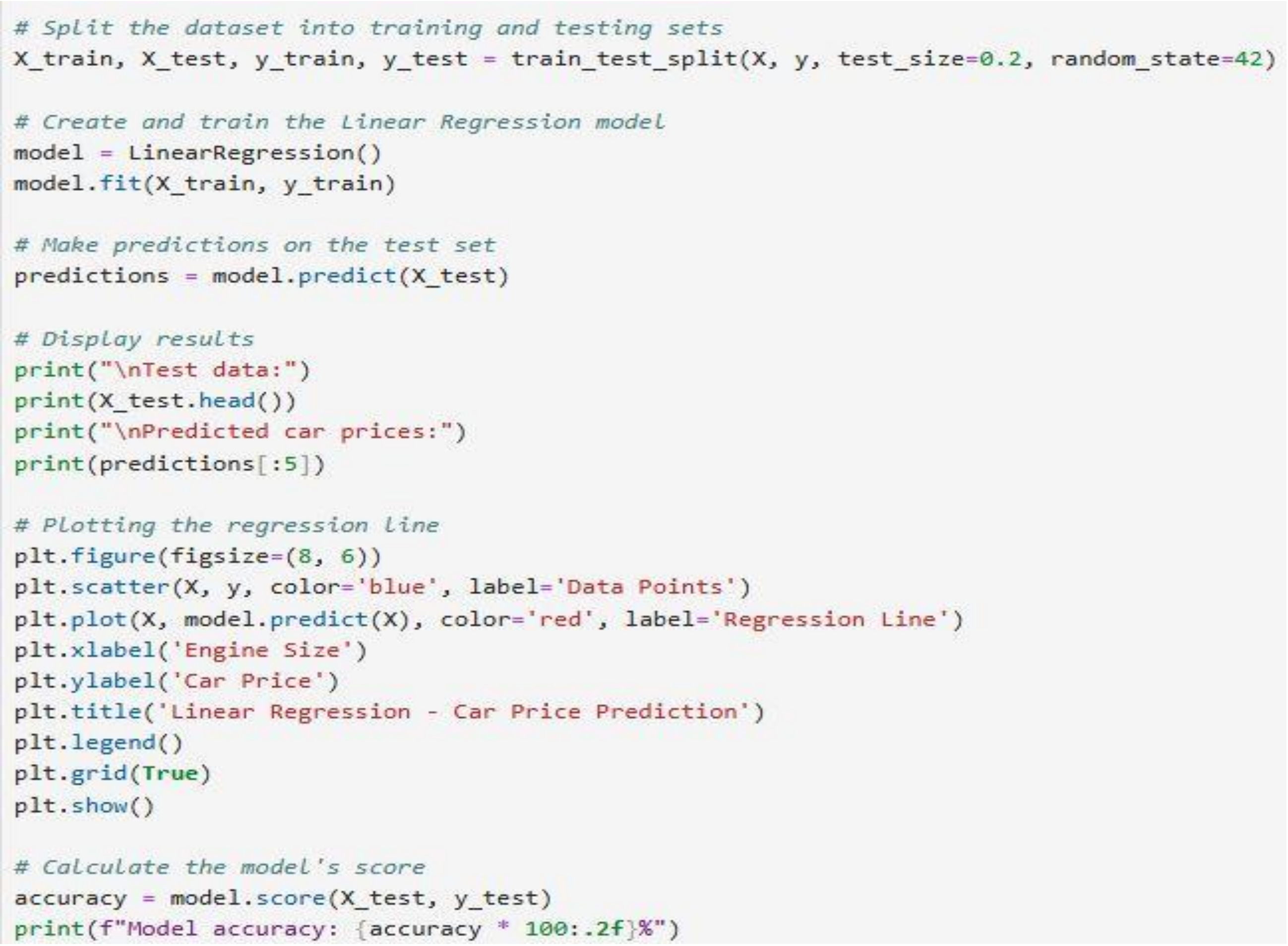
Home Tasks:

1. Implement a linear regression algorithm on any dataset to predict car prices based on their engine size using the dataset (`carprice.csv`).

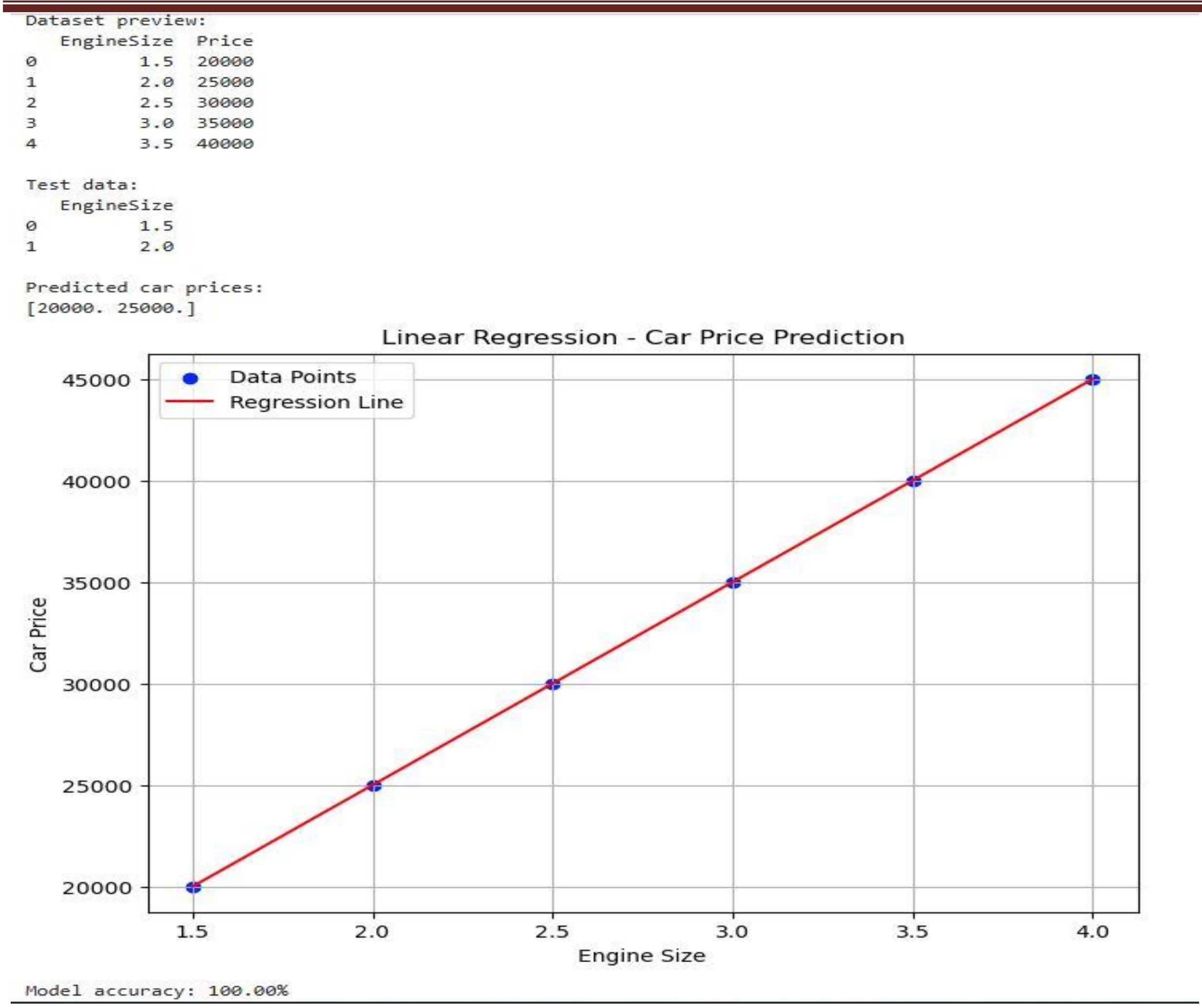
Engine Size (L)	Car Price ($)
1.5	20,000
2	25,000
2.5	30,000
3	35,000
3.5	40,000
4	45,000



```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

# Initialize the variable for dataset
data = None

# Try loading the dataset
try:
    data = pd.read_csv('carprice.csv')
except FileNotFoundError:
    print("Error: 'carprice.csv' file not found. Using fallback example dataset.")
    data = pd.DataFrame({
        'EngineSize': [1.5, 2.0, 2.5, 3.0, 3.5, 4.0],
        'Price': [20000, 25000, 30000, 35000, 40000, 45000]
    })

# Check if the dataset is valid
required_columns = ['EngineSize', 'Price']
if not all(col in data.columns for col in required_columns):
    print(f"Error: Dataset must contain the columns: {required_columns}")
    exit()

# Display the first few rows of the dataset
print("Dataset preview:")
print(data.head())

# Features and target variable
X = data[['EngineSize']]  # Independent variable
y = data['Price']         # Dependent variable

# Handle missing or invalid data
if data.isnull().sum().any():
    print("Error: Dataset contains missing values. Please clean the data before proceeding.")
    exit()
```



```
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train the Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
predictions = model.predict(X_test)

# Display results
print("\nTest data:")
print(X_test.head())
print("\nPredicted car prices:")
print(predictions[:5])

# Plotting the regression line
plt.figure(figsize=(8, 6))
plt.scatter(X, y, color='blue', label='Data Points')
plt.plot(X, model.predict(X), color='red', label='Regression Line')
plt.xlabel('Engine Size')
plt.ylabel('Car Price')
plt.title('Linear Regression - Car Price Prediction')
plt.legend()
plt.grid(True)
plt.show()

# Calculate the model's score
accuracy = model.score(X_test, y_test)
print(f"Model accuracy: {accuracy * 100:.2f}%")
```



```
Dataset preview:
   EngineSize  Price
0         1.5  20000
1         2.0  25000
2         2.5  30000
3         3.0  35000
4         3.5  40000

Test data:
   EngineSize
0         1.5
1         2.0

Predicted car prices:
[20000. 25000.]
```

```
Model accuracy: 100.00%
```

UploadedFileOnGitHub: