```
# Creating a list
fruits = ['apple', 'banana', 'cherry']



# Adding elements
fruits.append('orange')
fruits.insert(1, 'grapes')



# Slicing
print(fruits[1:3])  # ['grapes', 'cherry']
```

```
['grapes', 'banana']
```

```
# Looping through list
for fruit in fruits:
    print(fruit)
```

```
apple
grapes
banana
cherry
orange
```

```
import numpy as np


# Creating an array
arr = np.array([10, 20, 30, 40])



# Basic operations
print(arr + 5)         # [15 25 35 45]
print(arr * 2)         # [20 40 60 80]
```

```
[15 25 35 45]
[20 40 60 80]
```

```
# Indexing and slicing
print(arr[1:3])        # [20 30]
```

```
[20 30]
```

```
# 2D Array
matrix = np.array([[1, 2], [3, 4]])
print(matrix)
```

```
[[1 2]
 [3 4]]
```

```
import pandas as pd


# Creating a DataFrame
data = {
    'Name': ['Alice', 'Bob', 'Charlie'],
    'Marks': [85, 90, 95]
}
df = pd.DataFrame(data)


# Viewing data
print(df.head())
```

```
      Name  Marks
0    Alice     85
1      Bob     90
2  Charlie     95
```

```
# Adding a new column
df['Grade'] = ['A', 'A+', 'A+']


# Filtering rows
print(df[df['Marks'] > 85])
```

```
      Name  Marks Grade
1      Bob     90    A+
2  Charlie     95    A+
```

```
# Creating a dictionary
student = {
    'name': 'Alice',
    'age': 21,
    'course': 'Computer Science'
}
```

```
# Accessing and updating
print(student['name'])         # Output: Alice
student['age'] = 22            # Updating age
```

```
Alice
```

```
# Adding new key
student['college'] = 'MGM COE'
```

```
# Loop through dictionary
for key, value in student.items():
    print(f"{key}: {value}")
```

```
name: Alice
age: 22
course: Computer Science
college: MGM COE
```

```
# Importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
```

```
# --- NumPy Example ---
print("NumPy Array Operations")
arr = np.array([1, 2, 3, 4, 5])
print("Array:", arr)
print("Mean:", np.mean(arr))
print("Standard Deviation:", np.std(arr))
```

```
NumPy Array Operations
Array: [1 2 3 4 5]
Mean: 3.0
Standard Deviation: 1.4142135623730951
```

```
# --- Pandas Example ---
print("\nPandas DataFrame Example")
data = {'Name': ['Alice', 'Bob', 'Charlie'], 'Marks': [85, 90, 95]}
df = pd.DataFrame(data)
print(df)
print("Average Marks:", df['Marks'].mean())
```

```

Pandas DataFrame Example
      Name  Marks
0    Alice     85
1      Bob     90
2  Charlie     95
Average Marks: 90.0
```

```
# --- Scikit-Learn Example (Linear Regression) ---
print("\nScikit-Learn Linear Regression Example")
# Generating sample data
X = np.array([[1], [2], [3], [4], [5]])
y = np.array([1.2, 1.9, 3.2, 3.9, 5.1])
```

```

Scikit-Learn Linear Regression Example
```

```
# Splitting data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Creating and training model
model = LinearRegression()
model.fit(X_train, y_train)
```

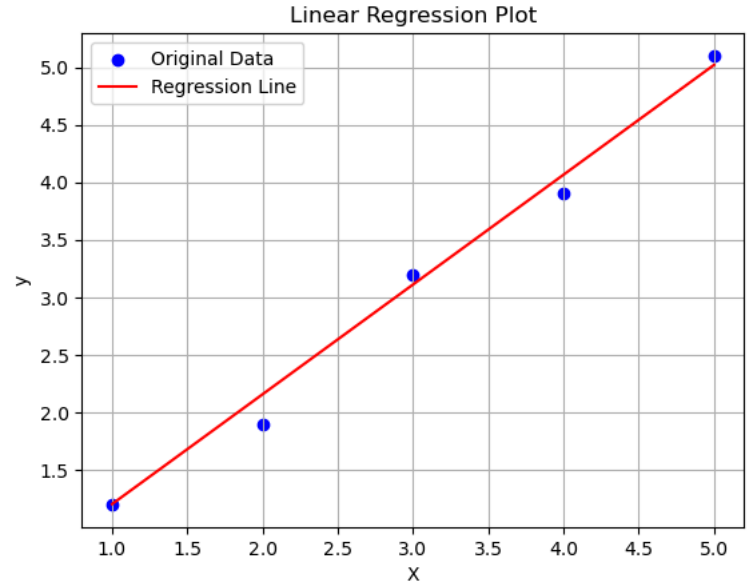
```
LinearRegression()
```

```
# Predicting
y_pred = model.predict(X_test)
print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
```

```
Mean Squared Error: 0.06612244897959205
```

```
# --- Matplotlib Example ---
print("\nMatplotlib Plot")
plt.scatter(X, y, color='blue', label='Original Data')
plt.plot(X, model.predict(X), color='red', label='Regression Line')
plt.title("Linear Regression Plot")
plt.xlabel("X")
plt.ylabel("y")
plt.legend()
plt.grid(True)
plt.show()
```

```
Matplotlib Plot
```




```
# Step 1: Import Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error


# Step 2: Load Dataset
# Sample dataset
data = pd.DataFrame({
    'Hours': [1, 2, 3, 4, 5, 6],
    'Scores': [10, 20, 30, 40, 50, 60]
})


# Step 3: Split features and target
X = data[['Hours']]  # input feature
y = data['Scores']   # target variable
```

```
# Step 4: Train-test split
# Step 5: Create and train the model
model = LinearRegression()
model.fit(X_train, y_train)
```

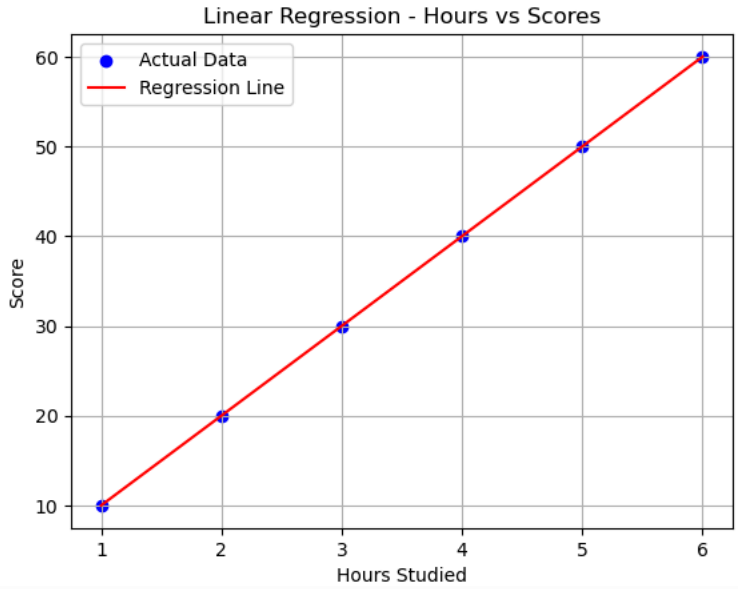
```
LinearRegression()
```

```
# Step 6: Prediction
y_pred = model.predict(X_test)


# Step 7: Evaluation
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
print("Slope (Coefficient):", model.coef_[0])
print("Intercept:", model.intercept_)
```

```
Mean Squared Error: 0.0
Slope (Coefficient): 10.0
Intercept: 0.0
```

```
# Step 8: Visualization
plt.scatter(X, y, color='blue', label='Actual Data')
plt.plot(X, model.predict(X), color='red', label='Regression Line')
plt.xlabel("Hours Studied")
plt.ylabel("Score")
plt.title("Linear Regression - Hours vs Scores")
plt.legend()
plt.grid(True)
plt.show()
```



Start coding or generate with AI.