Lab 3

Code

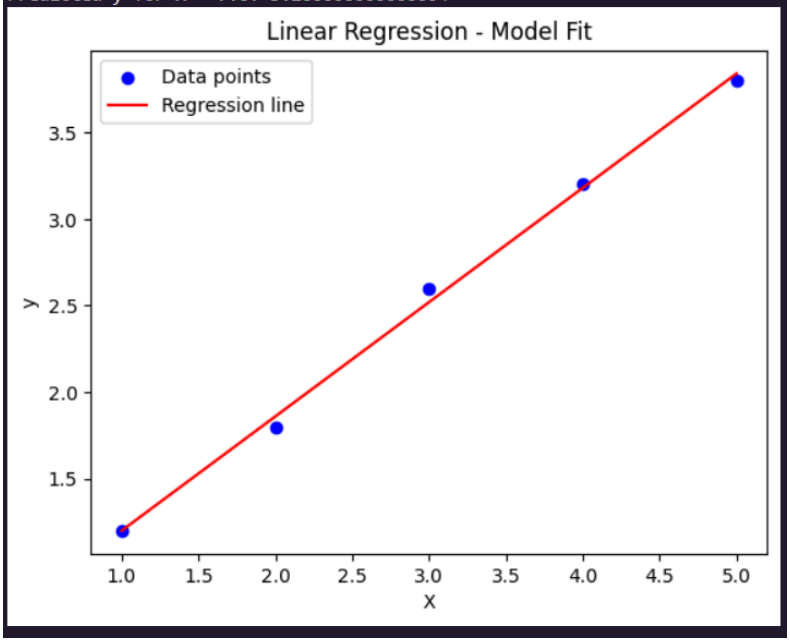
```
import numpy as np
import matplotlib.pyplot as plt

# Sample data (X and y)
X = np.array([1, 2, 3, 4, 5])  # Input features
y = np.array([1.2, 1.8, 2.6, 3.2, 3.8])  # Target variable

# Add a column of ones to X for the bias term (intercept)
X_b = np.c_[np.ones((X.shape[0], 1)), X]

# Calculate the coefficients (intercept and slope) using the Normal Equation
theta = np.linalg.inv(X_b.T.dot(X_b)).dot(X_b.T).dot(y)

# Intercept and Slope
intercept, slope = theta

# Print intercept and slope
print(f"Intercept (theta0): {intercept}")
print(f"Slope (theta1): {slope}")

# Make predictions for a new input value of X
x_input = float(input("Enter a value for X to predict y: "))
x_input_b = np.array([1, x_input])  # Add 1 for the bias term
y_pred = x_input_b.dot(theta)  # Calculate the predicted y

# Display the predicted value
print(f"Predicted y for X = {x_input}: {y_pred}")

# Visualize the data and the regression line
plt.scatter(X, y, color='blue', label='Data points')
plt.plot(X, X_b.dot(theta), color='red', label='Regression line')
plt.xlabel('X')
plt.ylabel('y')
plt.title('Linear Regression - Model Fit')
plt.legend()
plt.show()

import numpy as np
```

```
import matplotlib.pyplot as plt

# Sample data (X and y)
X = np.array([1, 2, 3, 4, 5])  # Input features
y = np.array([1.2, 1.8, 2.6, 3.2, 3.8])  # Target variable

# Initialize parameters
alpha = 0.01  # Learning rate
iterations = 1000  # Number of iterations
m = len(X)  # Number of training examples

theta0 = 0  # Intercept (bias)
theta1 = 0  # Slope

# Gradient Descent Algorithm
for _ in range(iterations):
    y_pred = theta0 + theta1 * X  # Predictions
    error = y_pred - y  # Compute error

    # Update parameters using gradient descent
    theta0 -= alpha * (1/m) * np.sum(error)
    theta1 -= alpha * (1/m) * np.sum(error * X)

# Print final intercept and slope
print(f"Intercept (theta0): {theta0}")
print(f"Slope (theta1): {theta1}")

# Make predictions for a new input value of X
x_input = float(input("Enter a value for X to predict y: "))
y_pred = theta0 + theta1 * x_input  # Calculate the predicted y

# Display the predicted value
print(f"Predicted y for X = {x_input}: {y_pred}")

# Visualize the data and the regression line
plt.scatter(X, y, color='blue', label='Data points')
plt.plot(X, theta0 + theta1 * X, color='red', label='Regression line')
plt.xlabel('X')
plt.ylabel('y')
plt.title('Linear Regression - Model Fit')
```

```
plt.legend()
plt.show()
```

Output

```
Intercept (theta0): 0.5400000000000025
Slope (theta1): 0.6600000000000001
Enter a value for X to predict y: 7
Predicted y for X = 7.0: 5.160000000000004
```




```
Intercept (theta0): 0.47890448451315504
Slope (theta1): 0.6769224781201147
Enter a value for X to predict y: 7
Predicted y for X = 7.0: 5.2173618313539585
```