**Name:** Muzammil Rauf Patel

**Class:** SYMCA **Div:** B

**Roll No:** 45 **Batch:** B2

**Course:** ML Lab

------------------------------------------------------------------------------------------------

**Assignment No: 3**

**Problem Statement:**

Implement Linear Regression on Dataset.

**------------------------------------------------------------------------------------------------**

**Code:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

# Load the dataset from a CSV file

data = pd.read\_csv('ass3.csv')

print(data)

# Assuming your dataset has two columns: 'X' for independent variable and 'y' for dependent variable

X = data['x'].values

print("Inependet Variable: ",X)

y = data['y'].values

print("Dependet Variable: ",y)

# Reshape X and y arrays to work with scikit-learn's LinearRegression

X = X.reshape(-1, 1)# Reshape X to a 2D array

y = y.reshape(-1, 1)

# Split the data 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 fit a Linear Regression model

model = LinearRegression()

model.fit(X\_train, y\_train)

print("Train dataset : ",X\_train)

# Get the coefficients (slope and intercept) of the linear regression model

slope = model.coef\_[0][0]

intercept = model.intercept\_[0]

print(f"Slope: {slope}")

print(f"Intercept: {intercept}")

# Make predictions on the test data

y\_pred = model.predict(X\_test)

#Plot the training data

plt.figure(figsize=(8, 6))

plt.scatter(X\_train, y\_train, color='blue', label='Training Data')

plt.plot(X\_train, model.predict(X\_train), color='red', linewidth=2, label='Linear Regression')

plt.title('Training Data and Linear Regression Model')

plt.xlabel('X')

plt.ylabel('y')

plt.legend()

plt.show()

# Plot the test data

plt.figure(figsize=(8, 6))

plt.scatter(X\_test, y\_test, color='blue', label='Test Data')

plt.plot(X\_test, y\_pred, color='red', linewidth=2, label='Predictions')

plt.title('Test Data and Predictions')

plt.xlabel('X')

plt.ylabel('y')

plt.legend()

plt.show()

#Take new input for prediction

new\_value = float(input("Enter a new value for prediction: "))

predicted\_y = model.predict(np.array([[new\_value]]))

print(f"Predicted y for X={new\_value}: {predicted\_y[0][0]}")

**Output:**