EXPERIMENT 1

Student Name: Samuel **UID: 24MAI10018**

Branch: CSE-AIML **Section/Group: 24MAI-1**

Semester: 2 **Date of Performance:**

Subject Name: Machine Learning Lab **Subject Code: 24CSH–651**

AIM:

Write a program in Python to implement Linear Regression Algorithm.

SOFTWARE REQUIREMENTS:

- Python IDE
- NumPy Library
- Jupyter Notebook

THEORY:

Supervised Learning: Supervised learning is a type of machine learning where a model is trained on a labeled dataset. The dataset consists of input-output pairs, where the input is the data (features) and the output is the corresponding label (target). The goal of supervised learning is to learn a mapping function that predicts the output for new, unseen inputs based on patterns in the training data.

Key features of Supervised Learning:

- Labeled Data: The training data has inputs and their corresponding correct outputs.
- Training Process: The algorithm learns by minimizing the error between predicted and actual outputs using a loss function.
- Prediction: Once trained, the model can predict outputs for new, unseen data.

Types of Supervised Learning:

- Regression: When the target output is continuous (e.g., predicting house prices, stock prices).
- Classification: When the target output is categorical (e.g., spam detection, image classification).

Regression: Regression in Supervised Learning is a task where the goal is to predict a continuous (numerical) output variable based on one or more input features. The relationship between the inputs (independent variables) and the output (dependent variable) is modelled, often as a mathematical function. Its key features include:

- Continuous Output: The target variable is numerical and can take any real value (e.g., temperature, sales, height, etc.).
- Feature-Target Relationship: Regression attempts to capture the underlying relationship between input features and the target variable.
- Minimizing Error: The model learns by reducing the difference (error) between predicted and actual values using a loss function (e.g., Mean Squared Error or Mean Absolute Error).

Linear Regression: Linear Regression is one of the simplest and most widely used algorithms in supervised learning. It is used to model the relationship between a dependent variable (target) and one or more independent variables (features) by fitting a linear equation to the data. Its key features are:

- Linear Relationship: Assumes a linear relationship between the input features and the target variable.
- Equation: The output is modelled as a linear function of the input: $y = mx + c$
  where, y: Predicted output
  x: Input Feature
  c: y-intercept (bias term)
  m: Slope of the line (weight of the feature)
- Objective: Minimize the error between predicted and actual values by finding the best-fit line. This is typically achieved by minimizing the Mean Squared Error (MSE).

ALGORITHM:

1. Initialize w and b randomly
2. Set the learning rate (alpha) and number of iterations
3. For each iteration:
   - Compute predictions: y_pred = w*x + b

- Compute the loss: Loss = (1/n) * sum((y_pred - y)^2)
- Compute gradients:
  - dw = -(2/n) * sum((y - y_pred) * x)
  - db = -(2/n) * sum(y - y_pred)
- Update parameters:
  - w = w - alpha * dw
  - b = b - alpha * db

4. Repeat until convergence

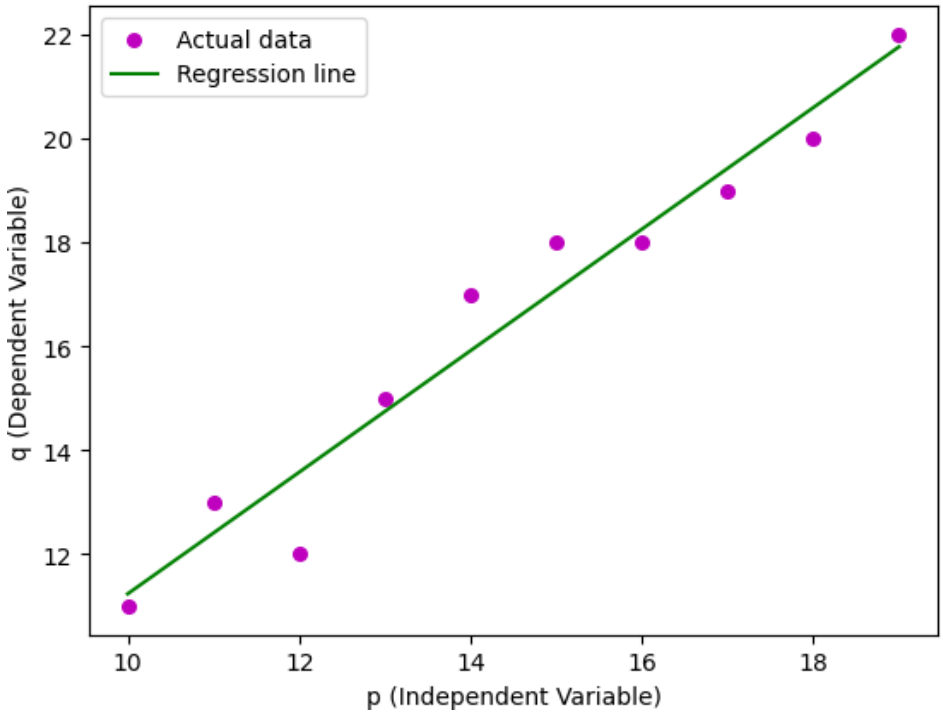
SOURCE CODE:

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

def estimate_coeff(p, q):

    n1 = np.size(p)
    m_p = np.mean(p)
    m_q = np.mean(q)


    SS_pq = np.sum(q * p) - n1 * m_q * m_p
    SS_pp = np.sum(p * p) - n1 * m_p * m_p


    b_1 = SS_pq / SS_pp
    b_0 = m_q - b_1 * m_p
    return (b_0, b_1)

def plot_regression_line(p, q, b):
    plt.scatter(p, q, color="m", marker="o", s=30, label="Actual data")
    q_pred = b[0] + b[1] * p
    plt.plot(p, q_pred, color="g", label="Regression line")
    plt.xlabel('p (Independent Variable)')
    plt.ylabel('q (Dependent Variable)')
    plt.legend()
    plt.show()

if __name__ == "__main__":
    p = np.array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19])
```

```
q = np.array([11, 13, 12, 15, 17, 18, 18, 19, 20, 22])

b = estimate_coeff(p, q)
print(f"Estimated coefficients are:")
print(f"b_0 = {b[0]:.2f} \nb_1 = {b[1]:.2f}\n")
plot_regression_line(p, q, b)
```

OUTPUT:

```
Estimated coefficients are :
b_0 = -0.46
b_1 = 1.17
```



LEARNING OUTCOMES:

1. Understood the basics of Linear Regression and its implementation.
2. Understood the task involving Data Pre-processing.
3. Gained the understanding of the mathematical concept of Linear Regression.
4. Learned the usage of Python Libraries for Linear Regression.
5. Learned Model Training and Evaluation for the same.