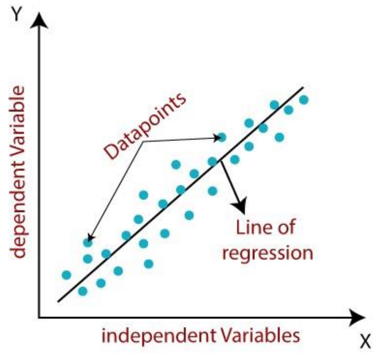
Machine Learning
Experiment 1

SAP ID: 60004200107 Name: Kartik Jolapara
Division: B Batch: B1

AIM
To implement Linear Regression.

THEORY
Linear regression is one of the easiest and most popular supervised Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, product price, etc.
Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.
The linear regression model provides a sloped straight line representing the relationship between the variables.



Mathematically, we can represent a linear regression as - y = w0 + w1 * x
Here, y = Dependent Variable (Target Variable) x = Independent Variable (Predictor Variable) w0 = Intercept of the line (Gives an additional degree of freedom) w1 = Linear regression coefficient (Scale factor to each input value).

Linear regression can be further divided into two types –
1. Simple Linear Regression

If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.

2. Multiple Linear regression

If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

CODE

Statistical Linear Regression

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

def estimate_coeff(x, y):
n = np.size(x)
mean_x = np.mean(x)
mean_y = np.mean(y)
   SS_xy = np.sum(y * x) - n * mean_y * mean_x
   SS_xx = np.sum(x * x) - n * mean_x * mean_x
   SS_yx2 = mean_y * np.sum(x * x) - mean_x * np.sum(x *
y)    SS_x = np.sum(x * x) - n * mean_x * mean_x    w_1 =
SS_xy / SS_xx    w_0 = SS_yx2 / SS_x    return (w_0, w_1)

def plot_regression_line(x, y, w):
   plt.scatter(x, y, color = "m", marker = "o", s =
30)    y_pred = w[0] + w[1] * x    plt.plot(x,
y_pred, color = "g")    plt.xlabel('x')
plt.ylabel('y')    plt.show()

def main():
   x = np.array([1, 2, 3, 4])    y = np.array([7, 3, 2, 4])    w =
estimate_coeff(x, y)    print("Estimated coefficients - \nw_0 = {}\nw_1 =
{}".format(w[0], w[1]))    print("The equation is : y = {} +
{}x\n".format(w[0], w[1]))    plot_regression_line(x, y, w)

if __name__ == "__main__":
   main()
```

Linear Regression using ML import

```
pandas as pd import numpy as np
import matplotlib.pyplot as plt data =
pd.read_csv('Salary_Data.csv')

x = data.iloc[:, 0]
y = data.iloc[:, -1]
```

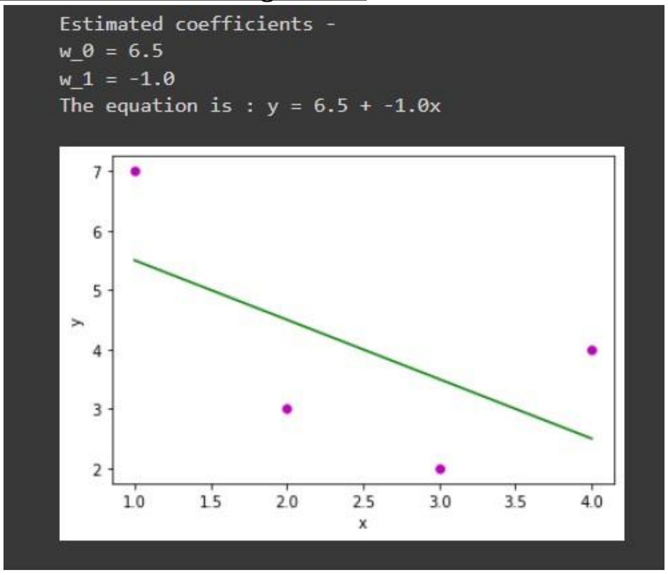
```
w_0 = 0.1
w_1 = 0.2
alpha = 0.01
epoch = 0
sumofdiff = 0
n = len(x)

for epoch in range (1000):
y1 = []   for i in x:
    y1.append(w_0 + w_1 * i)
for i in range(n):
    sumofdiff = sumofdiff + (y1[i] - y[i]) *
x[i]   delta = alpha * sumofdiff / n   w_0 =
w_0 - delta   w_1 = w_1 - delta   y = y1
print('w_0 = ', w_0) print('w_1 = ', w_1)

plt.scatter(x, y, color = "m", marker = "o", s =
30) plt.plot(x, y, color = "g") plt.xlabel('Years
Experience') plt.ylabel('Salary')
```

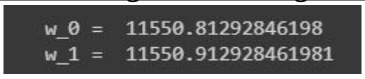
OUTPUT

<u>Statistical Linear Regression</u>

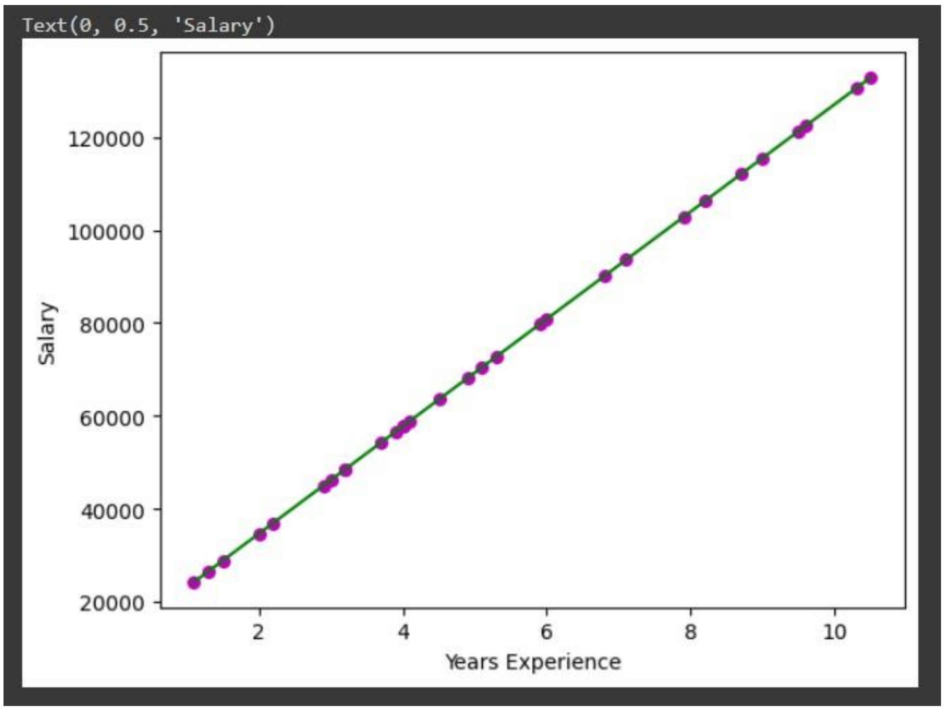


```
Estimated coefficients -
w_0 = 6.5
w_1 = -1.0
The equation is : y = 6.5 + -1.0x
```

Linear Regression using ML



```
w_0 =  11550.81292846198
w_1 =  11550.912928461981
```



```
Text(0, 0.5, 'Salary')
```

CONCLUSION

Thus, we have successfully implemented Linear Regression using both the statistical method and machine learning.