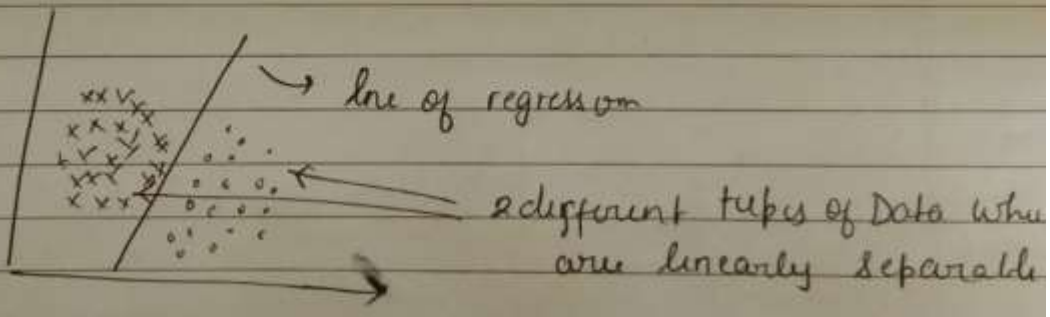
60004210210
Amartya Mishra
COMPS C31

ML- Experiment 1

Aim: To implement linear Regression

Theory:
LR is one of easiest & most popular ML Algos. It is statistical Method used for predictive analysis. It makes prediction for continous / real / numeric Variables such as sales, salary, age etc. It shows linear relationship between dependent (Y) & one ~~of~~ or more independent Variable (X)
It finds how values of x causes a change in the output variable Y
This Model provides sloped line representing relationship between variable.



Mathematically it is represented as

$$y = w_0 + w_1 x$$

y = in dependent Variable
w_0 = Y intercept
w_1 = Slope of line
x = Independent Variable.

FOR EDUCATIONAL USE

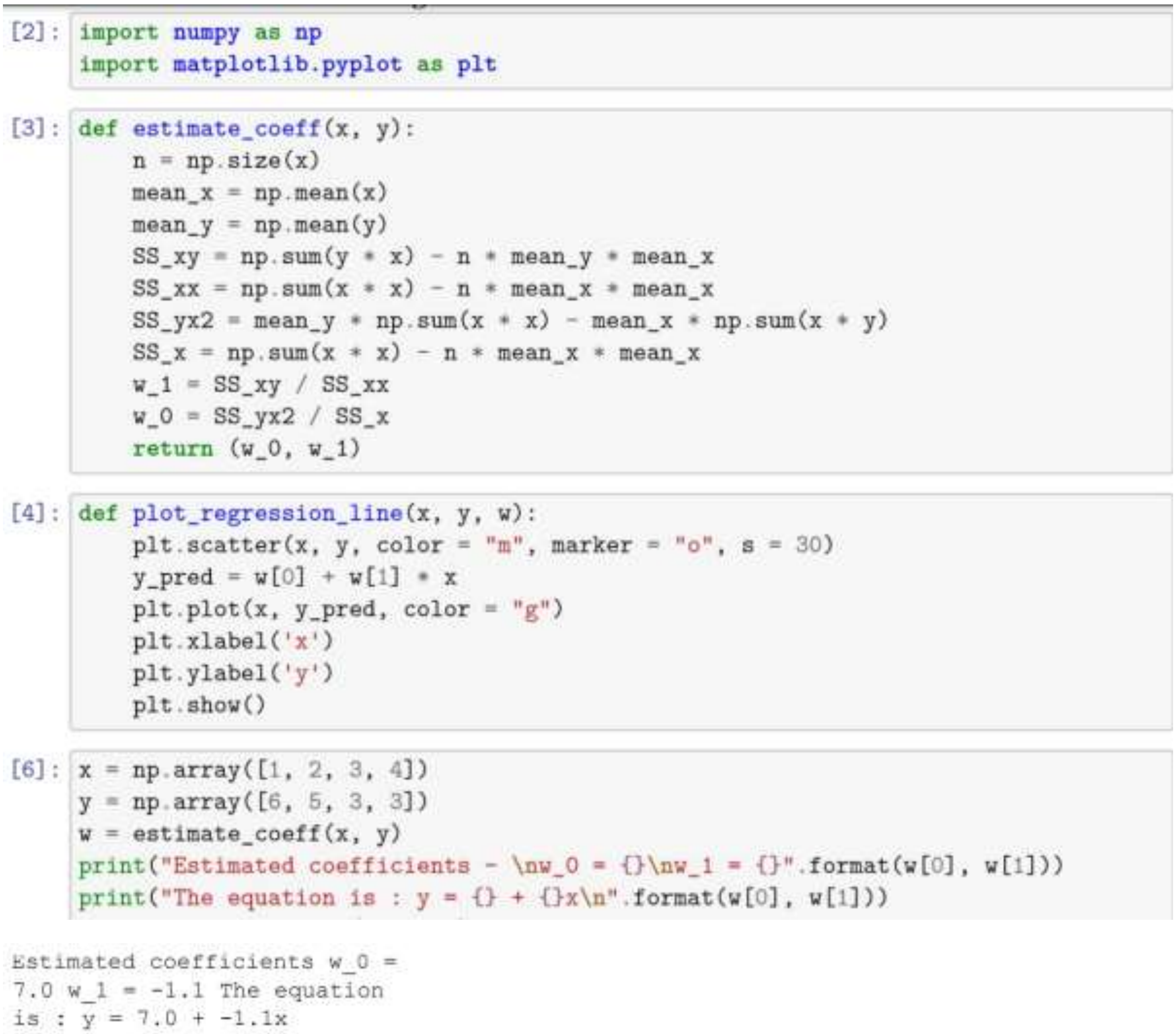
Linear Regression

- Simple - Single Independent Variable is used
- Multiple - More than one independent Variable used

- Conclusion

Hence we have implemented linear regression using both statistical Method & machine learning

Ml- Lab 1 Linear regression



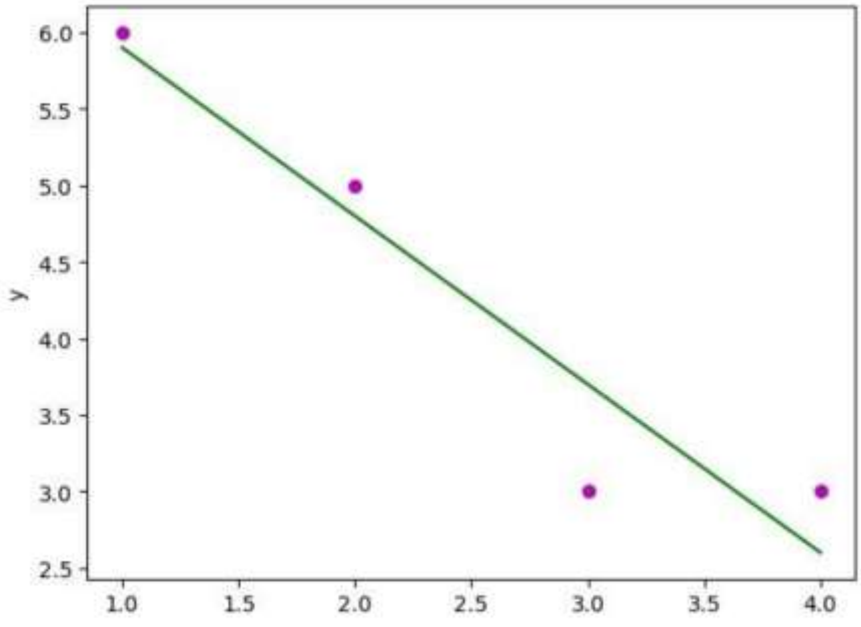
```
[2]: import numpy as np
     import matplotlib.pyplot as plt
```

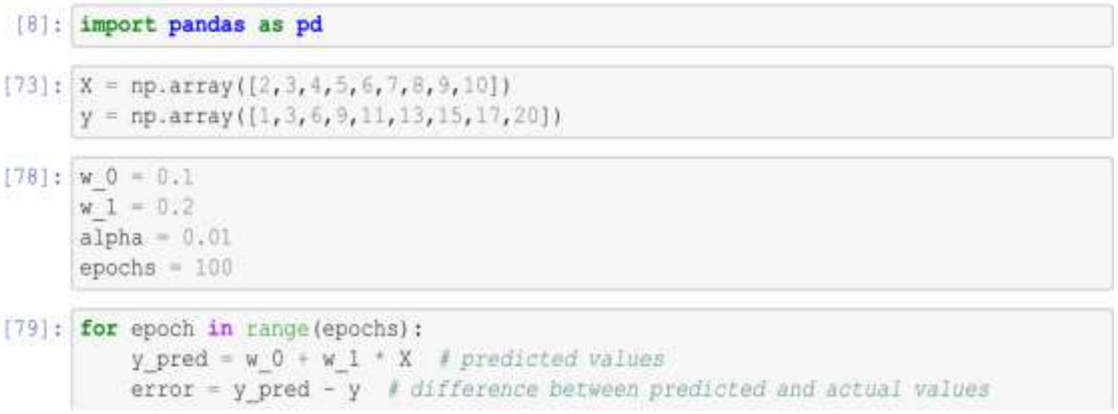
```
[3]: def estimate_coeff(x, y):
         n = np.size(x)
         mean_x = np.mean(x)
         mean_y = np.mean(y)
         SS_xy = np.sum(y * x) - n * mean_y * mean_x
         SS_xx = np.sum(x * x) - n * mean_x * mean_x
         SS_yx2 = mean_y * np.sum(x * x) - mean_x * np.sum(x * y)
         SS_x = 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)
```

```
[4]: 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()
```

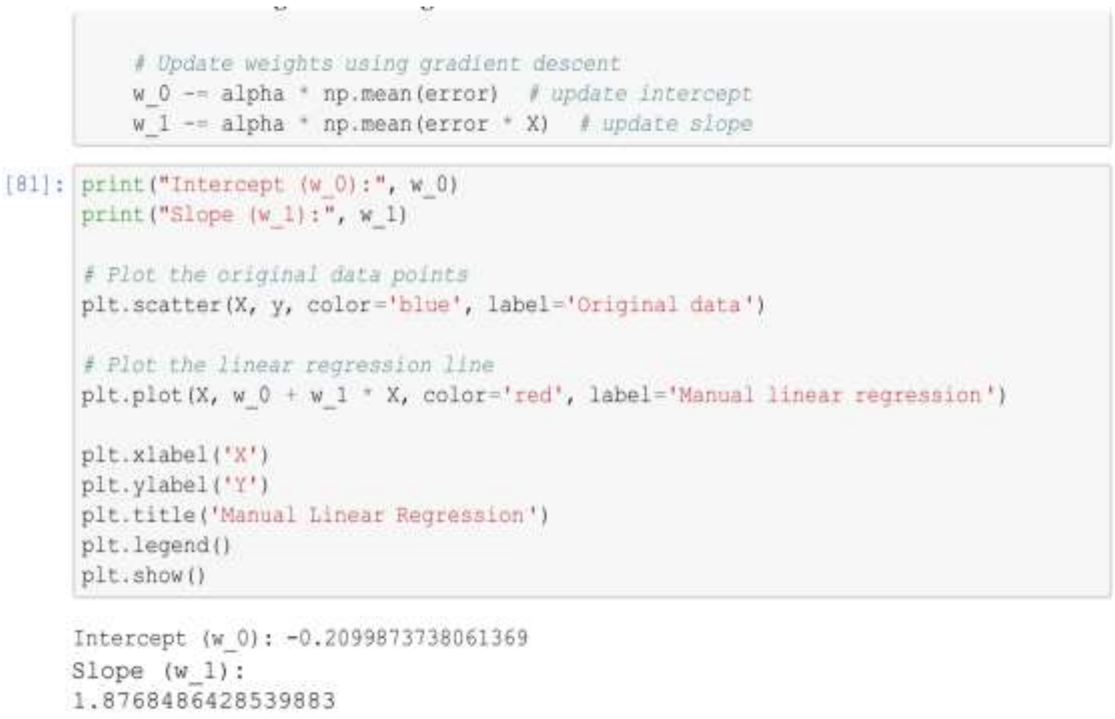
```
[6]: x = np.array([1, 2, 3, 4])
     y = np.array([6, 5, 3, 3])
     w = estimate_coeff(x, y)
     print("Estimated coefficients - \nw_0 = {}\nw_1 = {}".format(w[0], w[1]))
     print("The equation is : y = {} + {}x\n".format(w[0], w[1]))
```

```
Estimated coefficients w_0 =
7.0 w_1 = -1.1 The equation
is : y = 7.0 + -1.1x
```

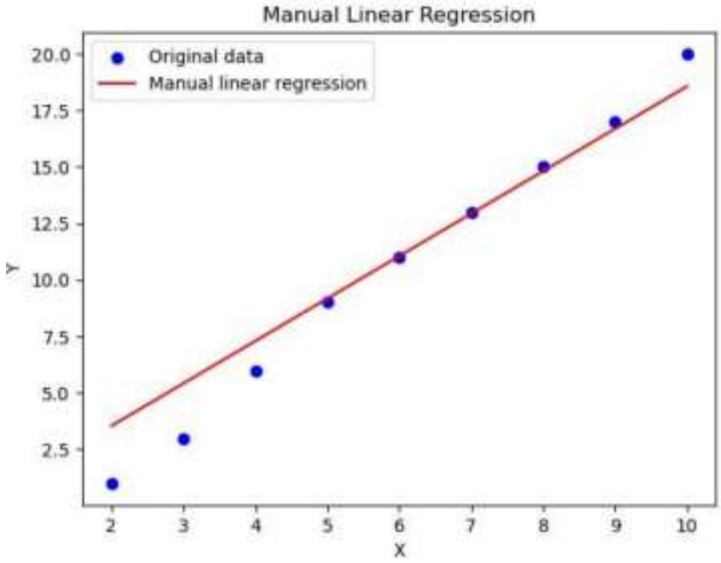




```
[8]: import pandas as pd

[73]: X = np.array([2,3,4,5,6,7,8,9,10])
      y = np.array([1,3,6,9,11,13,15,17,20])

[78]: w_0 = 0.1
      w_1 = 0.2
      alpha = 0.01
      epochs = 100

[79]: for epoch in range(epochs):
          y_pred = w_0 + w_1 * X  # predicted values
          error = y_pred - y  # difference between predicted and actual values
```



```
          # Update weights using gradient descent
          w_0 -= alpha * np.mean(error)  # update intercept
          w_1 -= alpha * np.mean(error * X)  # update slope

[81]: print("Intercept (w_0):", w_0)
      print("Slope (w_1):", w_1)

      # Plot the original data points
      plt.scatter(X, y, color='blue', label='Original data')

      # Plot the linear regression line
      plt.plot(X, w_0 + w_1 * X, color='red', label='Manual linear regression')

      plt.xlabel('X')
      plt.ylabel('Y')
      plt.title('Manual Linear Regression')
      plt.legend()
      plt.show()

      Intercept (w_0): -0.2099873738061369
      Slope (w_1):
      1.8768486428539883
```


Manual Linear Regression
Original data
Manual linear regression
Y
X