Name: Ankit Sharma

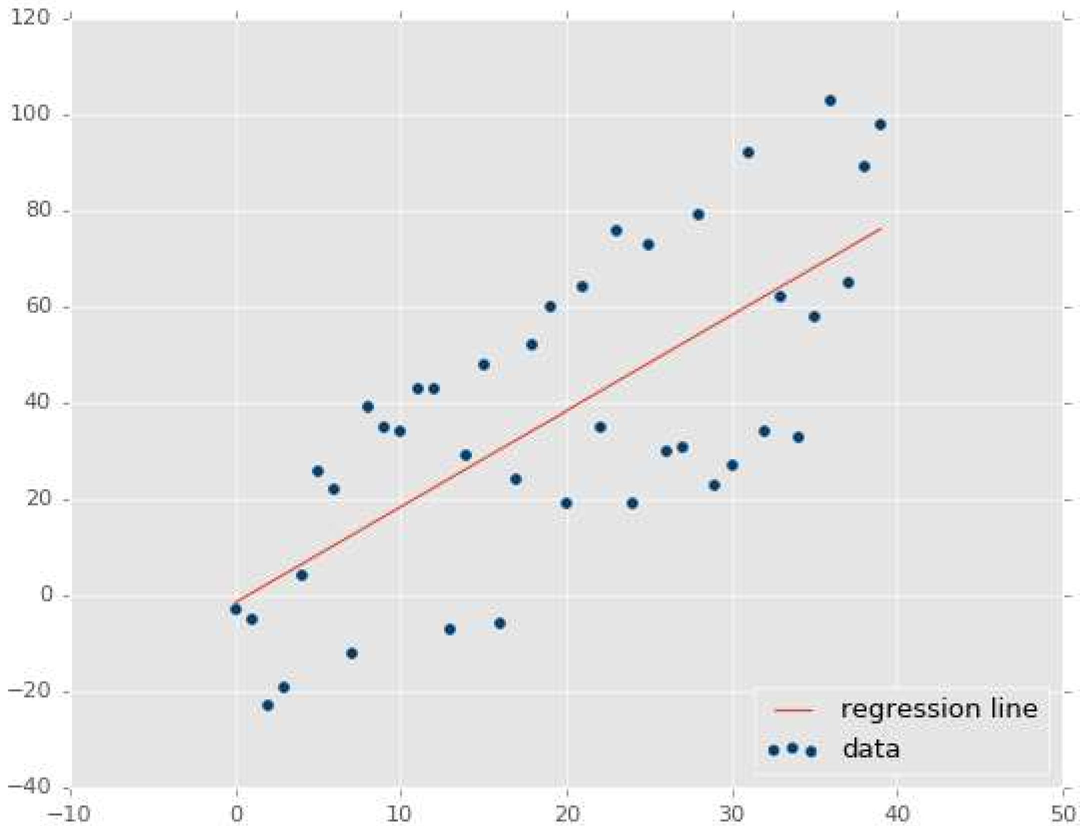
Email: kumarankitx022@gmail.com

Mob no.: +91-7677241423

ASSIGNMENT – 04

LINEAR REGRESSION MODEL OF SKLEARN:

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y (output). Hence, the name is Linear Regression. If we plot the independent variable (x) on the x-axis and dependent variable (y) on the y-axis, linear regression gives us a straight line that best fits the data points, as shown in the figure below.



We know that the equation of a straight line is basically: Y = mx + c

Where 'c' is the intercept and 'm' is the slope of the line. So basically, the linear regression algorithm gives us the most optimal value for the intercept and the slope (in two dimensions). The y and x variables remain the same, since they are the data features and cannot be changed.

The values that we can control are the intercept (c) and slope (m). There can be multiple straight lines depending upon the values of intercept and slope. Basically what the linear regression algorithm does is it fits multiple lines on the data points and returns the line that results in the least error.

This same concept can be extended to cases where there are more than two variables. This is called multiple linear regression. For instance, consider a scenario where you have to predict the price of the house based upon its area, number of bedrooms, the average income of the people in the area, the age of the house, and so on. In this case, the dependent variable (target variable) is dependent upon several independent variables. A regression model involving multiple variables can be represented as:

Y = m1x1 + m2x2 + m3x3 + ...+ mnxn + c

This is the equation of a hyperplane. Remember, a linear regression model in two dimensions is a straight line; in three dimensions it is a plane, and in more than three dimensions, a hyperplane.

Example:

```
>>> import numpy as np
>>> from sklearn.linear_model import LinearRegression
>>> X = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])
array([[1, 1],
       [1, 2],
       [2, 2],
       [2, 3]])
>>> y = np.dot(X, np.array([1, 2])) + 3
array([ 6,  8,  9, 11])
>>> reg = LinearRegression().fit(X, y)
>>> reg.score(X, y)
1.0
>>> reg.coef_
array([1., 2.])
>>> reg.intercept_
3.0000000000000018
>>> reg.predict(np.array([[3, 5]]))
array([16.])
```

Checking the values of Coefficients and Intercept based on different random_state

In [12]:

```
import random as rd
reg = linear_model.LinearRegression()
i = rd.sample(range(50),10)
x,a,b = [],[],[]
for e in i:
    X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.25, random_state
    reg.fit(X_train,Y_train)
    Y_pred = reg.predict(X_test)
    x.append(e)
    a.append(reg.coef_.round(decimals = 2))
    b.append(reg.intercept_.round(decimals = 2))
    rand_check = pd.DataFrame({
        'random_state': np.array(x).flatten(),
        'Coefficients': np.array(a).flatten(),
        'Intercept': np.array(b).flatten(),
    })
rand_check
```

Out[12]:

	random_state	Coefficients	Intercept
0	5	9337.14	27048.91
1	26	9532.38	26073.12
2	20	9621.99	25747.08
3	29	9452.90	26034.89
4	2	9518.15	23866.27
5	31	9554.99	25021.98
6	21	9349.63	24602.12
7	14	9778.56	24345.06
8	25	9307.12	27066.13
9	36	9487.41	25982.16