NAME: MALLIKARJUN S NANDYAL

SECTION: 5ISE2

ROLL NUMBER:20191ISE0093

```
import numpy as np

import matplotlib.pyplot as plt



def estimate_coef(x, y):

    # number of observations/points

    n = np.size(x)



    # mean of x and y vector

    m_x = np.mean(x)

    m_y = np.mean(y)



    # calculating cross-deviation and deviation about x

    SS_xy = np.sum(y*x) - n*m_y*m_x

    SS_xx = np.sum(x*x) - n*m_x*m_x
```

```
    # calculating regression coefficients

    b_1 = SS_xy / SS_xx

    b_0 = m_y - b_1*m_x



    return (b_0, b_1)



def plot_regression_line(x, y, b):

    # plotting the actual points as scatter plot

    plt.scatter(x, y, color = "m",

               marker = "o", s = 30)



    # predicted response vector

    y_pred = b[0] + b[1]*x



    # plotting the regression line

    plt.plot(x, y_pred, color = "g")
```

```
    # putting labels
    plt.xlabel('x')
    plt.ylabel('y')


    # function to show plot
    plt.show()


def main():
    # observations / data
    x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
    y = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12])


    # estimating coefficients
    b = estimate_coef(x, y)
    print("Estimated coefficients:\nb_0 = {}  \
          \nb_1 = {}".format(b[0], b[1]))
```

```
    # plotting regression line
    plot_regression_line(x, y, b)

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

Simple Linear Regression

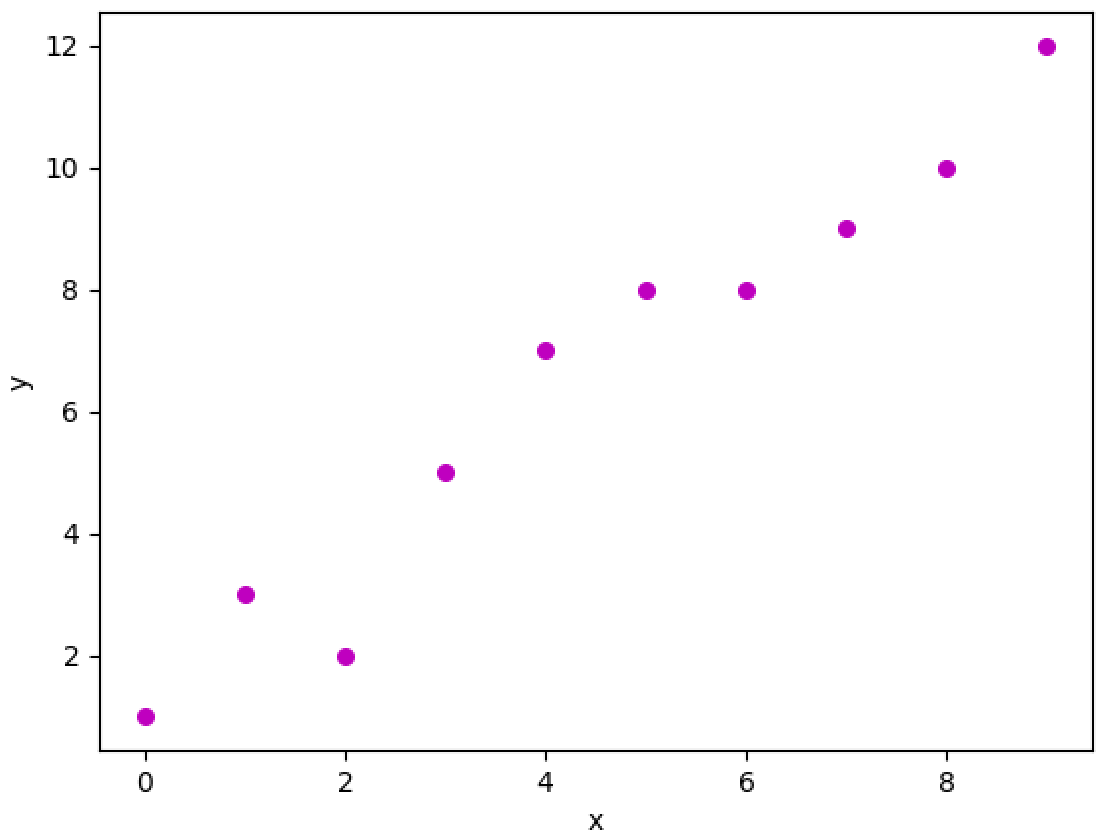
Simple linear regression is an approach for predicting a response using a single feature.

It is assumed that the two variables are linearly related. Hence, we try to find a linear function that predicts the response value(y) as accurately as possible as a function of the feature or independent variable(x).

Let us consider a dataset where we have a value of response y for every feature x:

x	0	1	2	3	4	5	6	7
y	1	3	2	5	7	8	8	9

For generality, we define:
x as feature vector, i.e x = [x_1, x_2, …., x_n],
y as response vector, i.e y = [y_1, y_2, …., y_n]
for n observations (in above example, n=10).
A scatter plot of the above dataset looks like:-



- h(x_i) represents the **predicted response value** for i^{th} observation.
- b_0 and b_1 are regression coefficients and represent **y-intercept** and **slope** of regression line respectively.

To create our model, we must “learn” or estimate the values of regression coefficients b_0 and b_1. And once we’ve estimated these coefficients, we can use the model to predict responses!
In this article, we are going to use the principle of **Least Squares**.
Now consider:

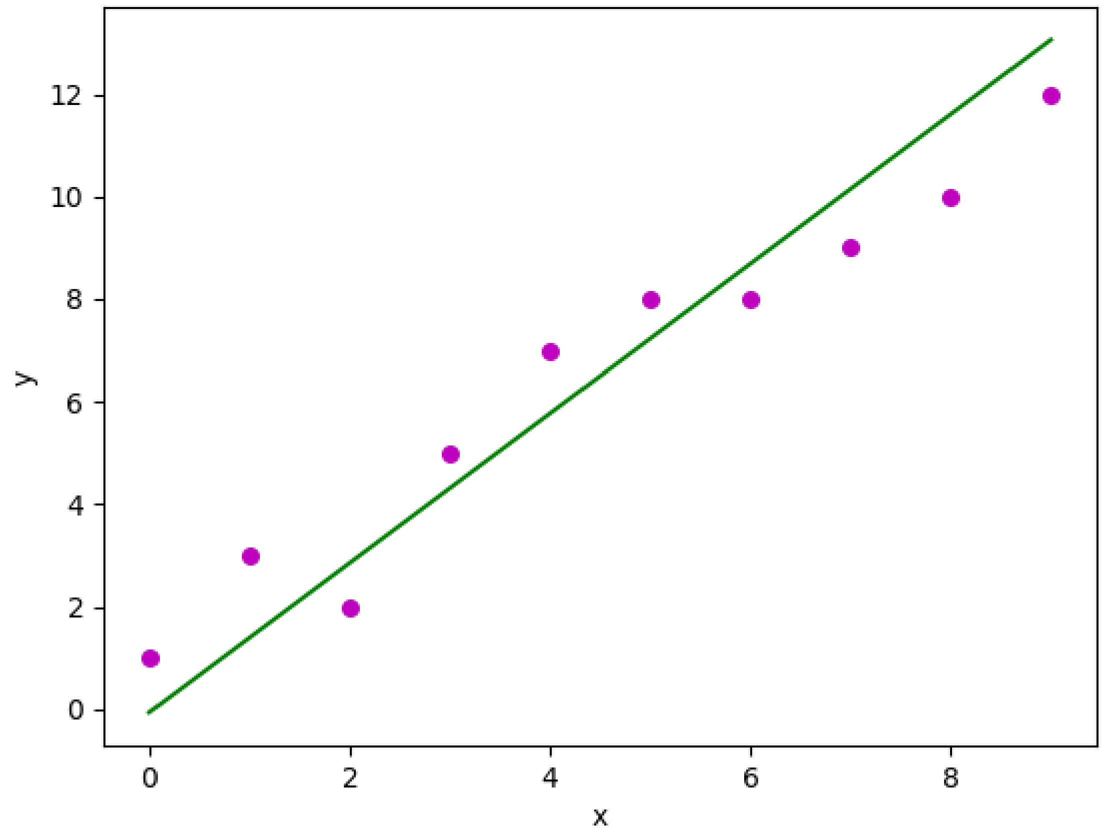
Here, e_i is a **residual error** in ith observation.
So, our aim is to minimize the total residual error.
We define the squared error or cost function, J as:

and our task is to find the value of b_0 and b_1 for which J(b_0,b_1) is minimum!
Without going into the mathematical details, we present the result here:

where SS_xy is the sum of cross-deviations of y and x:

and SS_xx is the sum of squared deviations of x:

Note: The complete derivation for finding least squares estimates in simple linear regression can be found here.



12
10
8
6
4
2
0
y
0
2
4
6
8
x