1. **Aim:** Write a python program to find the best fit straight line and draw the scatter plot.

**Source Code:**

import numpy as np

import matplotlib.pyplot as plt

x = np.array(list(map(float,input("Enter the x : ").split())))

y = np.array(list(map(float,input("Enter the y : ").split())))

print(x,y)

n = len(x)

tx = sum(x)

ty = sum(y)

tx2 = sum(x\*\*2)

txy = sum(x\*y)

a = ((tx2\*ty)-(tx\*txy))/(n\*tx2-(tx)\*\*2)

b = (ty - n\*a)/tx

yexp = a+b\*x

print(f'Eqn is y = {a}+{b}x')

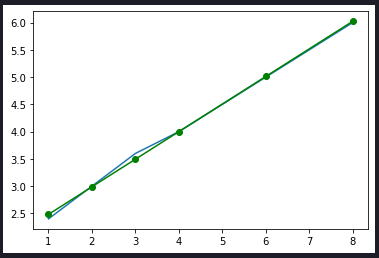
plt.plot(x,y)

plt.plot(x,yexp,color = 'Green',marker = 'o')

**In/Output:**

[1. 2. 3. 4. 6. 8.] [2.4 3. 3.6 4. 5. 6. ]

Eqn is y = 1.9764705882352933+0.5058823529411767x



1. **Aim:** Write a python program to fit a second degree parabola of the form y=a+bx+cx2 and draw the scatter plot.

**Source Code:**

import numpy as np

import pandas as pd

def getMatrixMinor(arr,i,j):

    c = arr[:]

    c = np.delete(c, (i),axis=0)

    return [np.delete(row, (j),axis=0) for row in (c)]

def determinant\_matrix(m):

    #base case for 2x2 matrix

    if len(m) == 2:

        return m[0][0]\*m[1][1]-m[0][1]\*m[1][0]

    deter = 0

    for c in range(len(m)):

        deter += ((-1)\*\*c)\*m[0][c]\*determinant\_matrix(getMatrixMinor(m,0,c))

    return deter

# Y=A+B.X+C.X^2

x=list(map(float,input("enter Values for X: ").split()))

print("X is:",x)

y=list(map(float,input("enter Values for Y: ").split()))

print("Y is:",y)

df=pd.DataFrame()

df['x']=x

df['y']=y

# xy=[x[i]\*y[i] for i in range(len(x))]

# x2y=[(x[i]\*\*2)\*y[i] for i in range(len(x))]

# x2=[x[i]\*\*2 for i in range(len(x))]

# x3=[x[i]\*\*3 for i in range(len(x))]

df['xy']=[x[i]\*y[i] for i in range(len(x))]

df['x2y']=[(x[i]\*\*2)\*y[i] for i in range(len(x))]

df['x2']=[x[i]\*\*2 for i in range(len(x))]

df['x3']=[x[i]\*\*3 for i in range(len(x))]

df['x4']=[x[i]\*\*4 for i in range(len(x))]

df

df.loc["total"] = df.sum()

df

# for i in df.columns:

#     df['x']['total']

eqn1=[df['y']['total'],len(x),df['x']['total'],df['x2']['total']]

eqn2=[df['xy']['total'],df['x']['total'],df['x2']['total'],df['x3']['total']]

eqn3=[df['x2y']['total'],df['x2']['total'],df['x3']['total'],df['x4']['total']]

eqn=[eqn1,eqn2,eqn3]

print(eqn1,eqn2,eqn3)

mat=np.matrix([eqn1[1:],eqn2[1:],eqn3[1:]])

mat=np.transpose(mat)

mat1=np.matrix([[eqn[i][0] for i in range(len(eqn))],eqn2[1:],eqn3[1:]])

mat1=np.transpose(mat1)

mat2=np.matrix([eqn1[1:],[eqn[i][0] for i in range(len(eqn))],eqn3[1:]])

mat2=np.transpose(mat2)

mat3=np.matrix([eqn1[1:],eqn2[1:],[eqn[i][0] for i in range(len(eqn))]])

mat3=np.transpose(mat3)

print(mat)

print(mat1)

print(mat2)

print(mat3)

det=np.linalg.det(mat)

det1=np.linalg.det(mat1)

det2=np.linalg.det(mat2)

det3=np.linalg.det(mat3)

solution=[det1/det,det2/det,det3/det]

solution

print("THE FITTED EQUATON IS:")

print("Y=",solution[0],"+",solution[1],"X +",solution[2],"X^2")

**In/Output:**

Enter Values for X: 2 4 6 8 10

X is: [2.0, 4.0, 6.0, 8.0, 10.0]

Enter Values for Y: 3.07 12.85 37.47 57.38 91.29

Y is: [3.07, 12.85, 37.47, 57.38, 91.29]

Out[ ]:

|  | **x** | **y** | **xy** | **x2y** | **x2** | **x3** | **x4** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 2.0 | 3.07 | 6.14 | 12.28 | 4.0 | 8.0 | 16.0 |
| **1** | 4.0 | 12.85 | 51.40 | 205.60 | 16.0 | 64.0 | 256.0 |
| **2** | 6.0 | 37.47 | 224.82 | 1348.92 | 36.0 | 216.0 | 1296.0 |
| **3** | 8.0 | 57.38 | 459.04 | 3672.32 | 64.0 | 512.0 | 4096.0 |
| **4** | 10.0 | 91.29 | 912.90 | 9129.00 | 100.0 | 1000.0 | 10000.0 |

1. Write a python program to find Karl Pearson’s correlation coefficient.
2. Write a python program to find the Spearman’s correlation coefficient between x and y variables.
3. Write a python program to classify the data based on one way Anova.
4. Write a python program to classify the data based on two way Anova.
5. Write a python program to fit a multiple regression model for any given data.
6. Write a python program to fit a multivariate regression model for any given data.
7. Write a python program to classify the treatments based on MANOVA Test.
8. Write a python program to classify the given observations using Linear Discriminant Analysis.
9. Write a python program to find Principle components for the given variables.
10. Write a python program to group the given variables using Factor Analysis.