NAME – SAARA ANAND

LINEAR REGRESSION IMPLEMENTATION-

CODE-

import matplotlib.pyplot as plt

import pandas as pd

from sklearn.model\_selection import train\_test\_split

dataset = pd.read\_csv('/content/Salary\_Data.csv')

X = dataset.iloc[:, :-1].values

y = dataset.iloc[:, -1].values

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 1/3, random\_state = 0)

from sklearn.linear\_model import LinearRegression

regressor = LinearRegression()

model=regressor.fit(X\_train, y\_train)

r\_sq = model.score(X\_train, y\_train)

print('coefficient of determination:', r\_sq)

print('intercept:', model.intercept\_)

print('slope:', model.coef\_)

y\_pred = model.predict(X\_test)

plt.scatter(X\_train, y\_train, color = 'red')

plt.plot(X\_train, regressor.predict(X\_train), color = 'BLUE')

plt.title('Salary vs Experience (Training set)')

plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.show()

plt.scatter(X\_test, y\_test, color = 'red')

plt.plot(X\_train, regressor.predict(X\_train), color = 'blue')

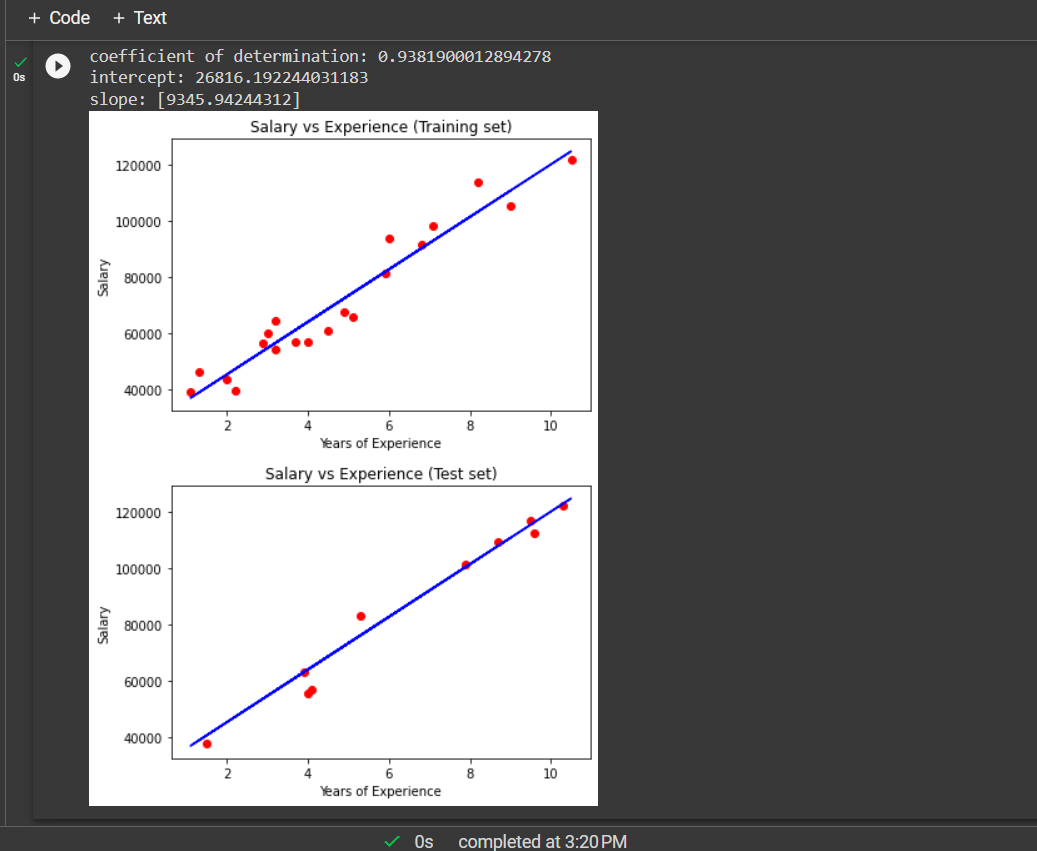
plt.title('Salary vs Experience (Test set)')

plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.show()

OUTPUT-



CODE-

import numpy as np

import matplotlib.pyplot as plt

def e\_coef(x, y):

  n = np.size(x)

  m\_x = np.mean(x)

  m\_y = np.mean(y)

  SS\_xy = np.sum(y\*x) - n\*m\_y\*m\_x

  SS\_xx = np.sum(x\*x) - n\*m\_x\*m\_x

  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):

  plt.scatter(x, y, color = "m",marker = "o", s = 30)

  y\_pred = b[0] + b[1]\*x

  plt.plot(x, y\_pred, color = "blue")

  plt.xlabel('x')

  plt.ylabel('y')

  plt.show()

def main():

  x = np.array([2011, 2012, 2013, 2014, 2015, 2016, 2017])

  y = np.array([12, 32, 42, 55, 67, 73, 80])

  b = e\_coef(x, y)

  print("Estimated coefficients:\nb\_0 = {} \\nb\_1 = {}".format(b[0], b[1]))

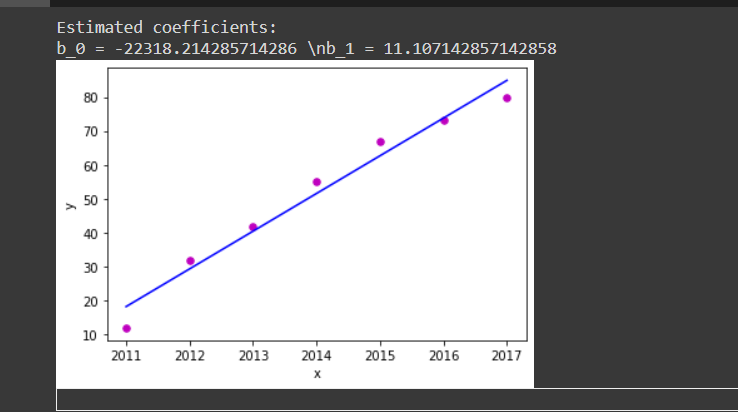
  plot\_regression\_line(x, y, b)

main()

x=input()

newsales=b[0]+b[1]\*x

OUTPUT-



CODE-

import numpy as nmp

import matplotlib.pyplot as mtplt

def estimate\_coeff(p, q):

    n1 = nmp.size(p)

    m\_p = nmp.mean(p)

    m\_q = nmp.mean(q)

    SS\_pq = nmp.sum(q \* p) - n1 \* m\_q \* m\_p

    SS\_pp = nmp.sum(p \* p) - n1 \* m\_p \* m\_p

    b\_1 = SS\_pq / SS\_pp

    b\_0 = m\_q - b\_1 \* m\_p

    return (b\_0, b\_1)

def plot\_regression\_line(p, q, b):

    mtplt.scatter(p, q, color = "m",

            marker = "o", s = 30)

    q\_pred = b[0] + b[1] \* p

    mtplt.plot(p, q\_pred, color = "g")

    mtplt.xlabel('p')

    mtplt.ylabel('q')

    mtplt.show()

def main():

    p = np.array([16, 45, 12, 3, 19, 15, 6, 19, 15, 19])

    q = np.array([1, 67, 12, 5, 17, 10, 18, 7, 28, 22])

# now, we will estimate the coefficients

    b = estimate\_coeff(p, q)

    print("Estimated coefficients are :\nb\_0 = {} \\nb\_1 = {}".format(b[0], b[1]))

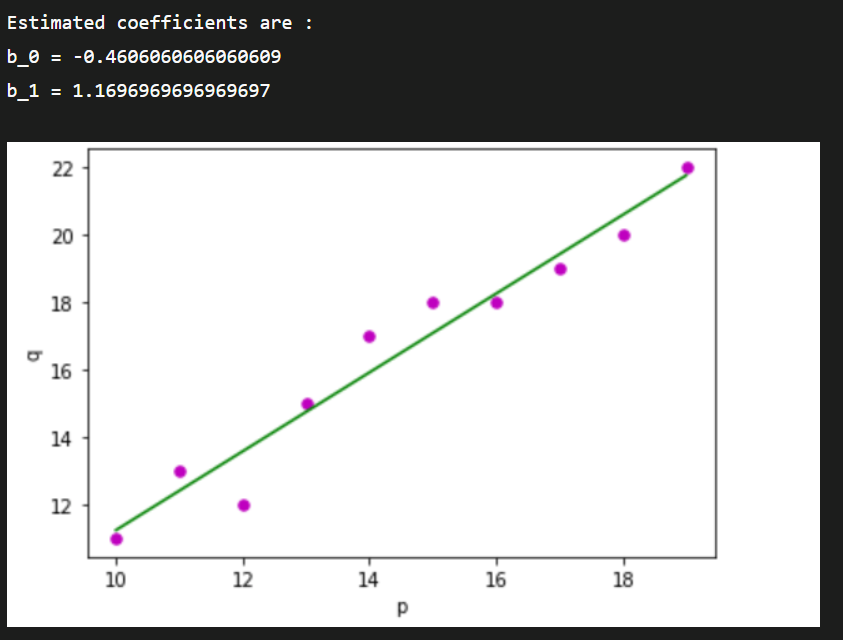
# Now, we will plot the regression line

    plot\_regression\_line(p, q, b)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

OUTPUT-



-----------------------------------------X---------------------------------------

Thank you!