**1.Consider the following observations/data. And apply simple linear regression and**

**find out estimated coefficients b0 and b1.( use numpypackage)**

**x=[0,1,2,3,4,5,6,7,8,9,11,13] y = ([1, 3, 2, 5, 7, 8, 8, 9, 10, 12,16, 18]**

import numpy as np

x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 13])

y = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12, 16, 18])

n = len(x)

mean\_x = np.mean(x)

mean\_y = np.mean(y)

b1 = np.sum((x - mean\_x) \* (y - mean\_y)) / np.sum((x - mean\_x)\*\*2)

b0 = mean\_y - b1 \* mean\_x

b0, b1

Output:

(0.838709677419355, 1.2889200561009817

**2..Consider the student data set. It can be downloaded from:**

**https://drive.google.com/open?id=1oakZCv7g3mlmCSdv9J8kdSaqO 5\_6dIOw . Write a**

**programme in python to apply simple linear regression and find out mean absolute error,**

**mean squared error and root mean squared error.**

import pandas as pd

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error

import numpy as np

data = pd.read\_csv('C:/Users/HP/Downloads/student\_scores.csv')

print(data.columns)

X = data[['Hours']]

y = data['Scores']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model = LinearRegression()

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

y\_pred = model.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = np.sqrt(mse)

mae, mse, rmse

Output:

Index(['Hours', 'Scores'], dtype='object')

(3.9207511902099244, 18.943211722315272, 4.352380006653288)

**3.Consider the following observations/data. And apply simple linear regression and**

**find out estimated coefficients b1 and b1 Also analyse theperformance of the model (Use**

**sklearn package) x = np.array([1,2,3,4,5,6,7,8]) y = np.array([7,14,15,18,19,21,26,23])**

pip install numpy scikit-learn

import numpy as np

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

x = np.array([1, 2, 3, 4, 5, 6, 7, 8]).reshape(-1, 1)

y = np.array([7, 14, 15, 18, 19, 21, 26, 23])

model = LinearRegression()

model.fit(x, y)

b1 = model.coef\_[0]

b0 = model.intercept\_

y\_pred = model.predict(x)

mse = mean\_squared\_error(y, y\_pred)

r2 = r2\_score(y, y\_pred)

b0, b1, mse, r2

Output:

(7.642857142857142, 2.2738095238095237, 3.4657738095238084, 0.8867741072947811)

**4.Write a python program to implement multiple Linear Regression modelfor a car**

**dataset. Dataset can be downloaded from:**

**https://github.com/krishnaik06/Multiple-Linear-Regression/blob/master/50\_Startups.csv**

Ans

import pandas as pd

import numpy as np

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

from sklearn.linear\_model import LinearRegression

from sklearn.preprocessing import LabelEncoder

url =

'https://raw.githubusercontent.com/krishnaik06/Multiple-Linear-Regression/master/50\_Startups.c

sv'

data = pd.read\_csv(url)

le = LabelEncoder()

data['State'] = le.fit\_transform(data['State'])

X = data.iloc[:, :-1].values

y = data.iloc[:, -1].values

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

regressor = LinearRegression()

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

y\_pred = regressor.predict(X\_test)

print(np.column\_stack((y\_pred, y\_test)))

Output:

[[103959.40508021 103282.38 ]

[132398.73236694 144259.4 ]

[133529.03790148 146121.95 ]

[ 72958.28368314 77798.83 ]

[179534.78737166 191050.39 ]

[115533.62584099 105008.31 ]

[ 67476.95847883 81229.06 ]

[ 98504.36199396 97483.56 ]

[114789.16083781 110352.25 ]

[168972.21909945 166187.94 ]]

**5.Write a python programme to implement multiple linear regression modelfor stock**

**market data frame as follows: Stock\_Market = {'Year':**

**[2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2016,2**

**016,20,16,2016,2016,2016,2016,2016,2016,2016,2016,2016], 'Month': [12,**

**11,10,9,8,7,6,5,4,3,2,1,12,11,10,9,8,7,6,5,4,3,2,1],**

**'Interest\_Rate': [2.75,2.5,2.5,2.5,2.5,2.5,2.5,2.25,2.25,2.25,2,2,2,1.75,1.75,1.75,1.75,1.75,1**

**.75,1.75,1.75,1.75,1.75,1.75],**

**'Unemployment\_Rate':**

**[5.3,5.3,5.3,5.3,5.4,5.6,5.5,5.5,5.5,5.6,5.7,5.9,6,5.9,5.8,6.1,6.2,6.1,6.1,6.1,5 .9,6.2,6.2,6.1],**

**'Stock\_Index\_Price':**

**[1464,1394,1357,1293,1256,1254,1234,1195,1159,1167,1130,1075,1047,**

**965,943,958,971,949,884,866,876,822,704,719] } And draw a graph of stock market price**

**verses interest rate.**

ANS:

pip install pandas numpy statsmodels matplotlib

import pandas as pd

import numpy as np

import statsmodels.api as sm

import matplotlib.pyplot as plt

Stock\_Market = {

'Year': [2017]\*12 + [2016]\*12,

'Month': list(range(12, 0, -1)) \* 2,

'Interest\_Rate': [

2.75, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.25, 2.25, 2.25, 2, 2,

2, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75

],

'Unemployment\_Rate': [

5.3, 5.3, 5.3, 5.3, 5.4, 5.6, 5.5, 5.5, 5.5, 5.6, 5.7, 5.9,

6, 5.9, 5.8, 6.1, 6.2, 6.1, 6.1, 6.1, 5.9, 6.2, 6.2, 6.1

],

'Stock\_Index\_Price': [

1464, 1394, 1357, 1293, 1256, 1254, 1234, 1195, 1159, 1167, 1130, 1075,

1047, 965, 943, 958, 971, 949, 884, 866, 876, 822, 704, 719

]

}

df = pd.DataFrame(Stock\_Market)

X = df[['Interest\_Rate', 'Unemployment\_Rate']]

y = df['Stock\_Index\_Price']

X = sm.add\_constant(X)

model = sm.OLS(y, X).fit()

print(model.summary())

plt.figure(figsize=(10, 6))

plt.scatter(df['Interest\_Rate'], df['Stock\_Index\_Price'], color='blue')

plt.title('Stock Market Price vs Interest Rate')

plt.xlabel('Interest Rate')

plt.ylabel('Stock Index Price')

plt.grid()

plt.show()

OUTPUT:

OLS Regression Results

==============================================================================

Dep. Variable: Stock\_Index\_Price R-squared: 0.898

Model: OLS Adj. R-squared: 0.888

Method: Least Squares F-statistic: 92.07

Date: Wed, 25 Sep 2024 Prob (F-statistic): 4.04e-11

Time: 00:47:38 Log-Likelihood: -134.61

No. Observations: 24 AIC: 275.2

Df Residuals: 21 BIC: 278.8

Df Model: 2

Covariance Type: nonrobust

================================================================================

=====

coef std err t P>|t| [0.025 0.975]

-------------------------------------------------------------------------------------

const 1798.4040 899.248 2.000 0.059 -71.685 3668.493

Interest\_Rate 345.5401 111.367 3.103 0.005 113.940 577.140

Unemployment\_Rate -250.1466 117.950 -2.121 0.046 -495.437 -4.856

==============================================================================

Omnibus: 2.691 Durbin-Watson: 0.530

Prob(Omnibus): 0.260 Jarque-Bera (JB): 1.551

Skew: -0.612 Prob(JB): 0.461

Kurtosis: 3.226 Cond. No. 394.

==============================================================================

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

**6. Consider the following observations/data. And apply simple linear regression and**

**find out estimated coefficients b1 and b1 Also analyse theperformance of the model (Use**

**sklearn package) x = np.array([1,2,3,4,5,6,7,8]) y = np.array([7,14,15,18,19,21,26,23])**

Ans

import numpy as np

import pandas as pd

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

import matplotlib.pyplot as plt

x = np.array([1, 2, 3, 4, 5, 6, 7, 8]).reshape(-1, 1)

y = np.array([7, 14, 15, 18, 19, 21, 26, 23])

model = LinearRegression()

model.fit(x, y)

b0 = model.intercept\_

b1 = model.coef\_[0]

y\_pred = model.predict(x)

mse = mean\_squared\_error(y, y\_pred)

r2 = r2\_score(y, y\_pred)

print(f'Estimated coefficients:')

print(f'b0 (intercept): {b0}')

print(f'b1 (slope): {b1}')

print(f'Mean Squared Error: {mse}')

print(f'R^2 Score: {r2}')

plt.scatter(x, y, color='blue', label='Data Points')

plt.plot(x, y\_pred, color='red', label='Regression Line')

plt.title('Simple Linear Regression')

plt.xlabel('X')

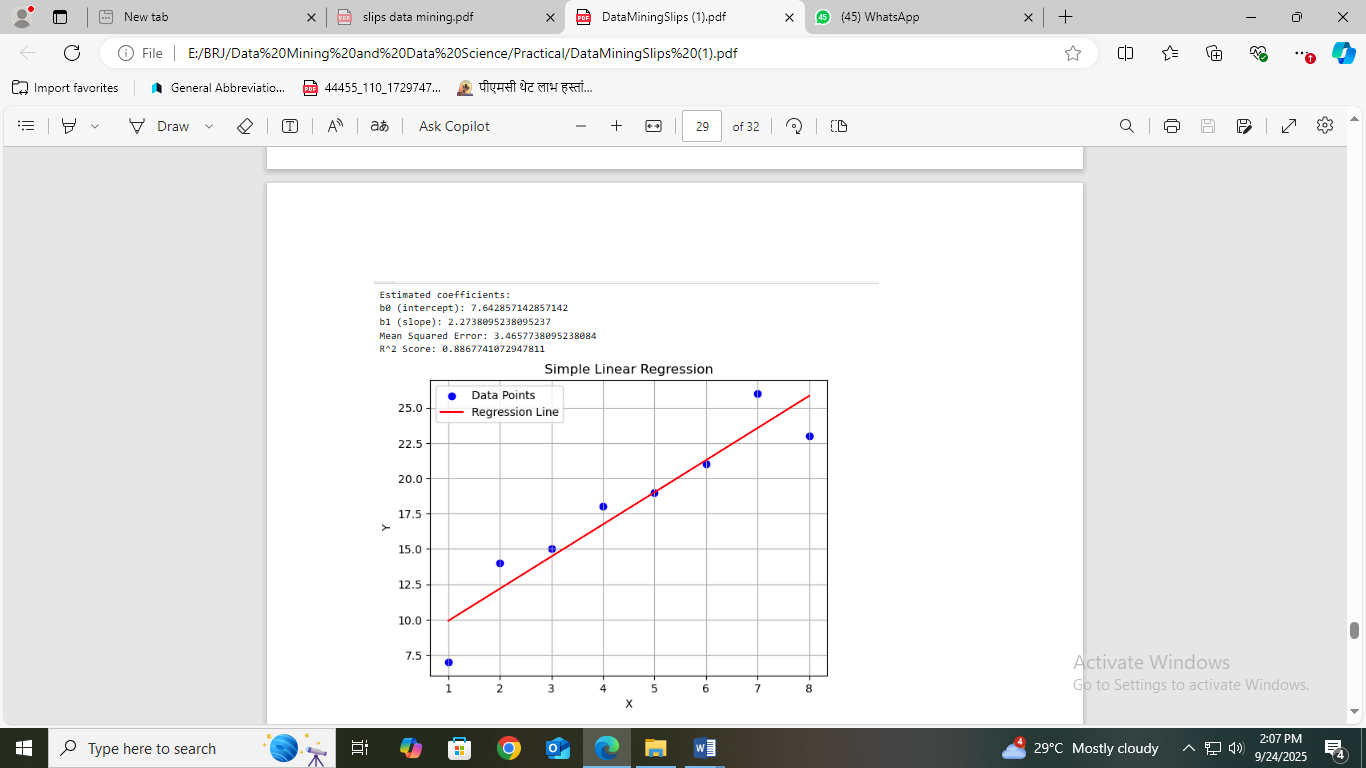
plt.ylabel('Y')

plt.legend()

plt.grid()

plt.show()

Output:



**7.Write a python program to implement multiple Linear Regression modelfor a car**

**dataset. Dataset can be downloaded from: (**

**https://www.w3schools.com/python/python\_ml\_multiple\_regression.asp )**

**( https://www.kaggle.com/datasets/swapnil1894/multiple-regression )**

Ans

import pandas as pd

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

# Load data from CSV

data = pd.read\_csv("C:/Users/HP/Downloads/1.02.Multiplelinearregression.csv")

# Check the columns in the DataFrame

print(data.columns)

# Replace 'SAT', 'Rand' with the correct feature columns if they differ

X = data[['SAT', 'Rand 1,2,3']]

y = data['GPA']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model = LinearRegression()

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

y\_pred = model.predict(X\_test)

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print("Mean Squared Error:", mse)

print("R-squared:", r2)

Output:

Index(['SAT', 'GPA', 'Rand 1,2,3'], dtype='object')

Mean Squared Error: 0.05766690113933144

R-squared: 0.417127123666891