

Exp. NO: 2

A: PYTHON PROGRAM TO IMPLEMENT

Date: 8/8/25.

SIMPLE LINEAR REGRESSION USING

LEAST SQUARE METHOD

Aim:

To implement a python program for constructing a simple linear regression using least square method.

Algorithm:

Step 1: Import necessary libraries:

\* pandas for data manipulation and matplotlib.pyplot for plotting

Step 2: Read the dataset:

\* Use the pandas 'read\_csv' function to read the dataset.

Step 3: Prepare the data:

\* Extract the independent variable (x) and dependent variable from the dataset.

\* Reshape x and y to be 2D arrays if needed.

Step 4: Calculate the mean:

\* Calculate the mean of x and y.

Step 5: Calculate the coefficients.

\* Calculate the slope (m) using the formula:

$$m = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

\* Calculate the intercept (b) using formula:  $b = \bar{y} - m\bar{x}$

Step 6: Make predictions:

\* Use the calculated slope and intercept to make prediction for each x value.

$$\hat{y} = mx + b$$



## Program :

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
data = pd.read_csv('headbrain.csv')
x, y = np.array(list(data['Head size (cm3)']),
                 np.array(list(data['Brain Weight (grams)'])))
Print(x[:5], y[:5])
[4512 3738 4261 3777 4177] [1530 1297 1335 1282 159]
def get_lines(x, y):
    x-m, y-m = np.mean(x), np.mean(y)
    Print(x-m, y-m)
    x-d, y-d = x-x-m, y-y-m
    m = np.sum(x-d * y-d) / np.sum(x-d ** 2)
    c = y-m - (m * x-m)
    Print(m, c)
    return lambda k: m * k + c
lin = get_line(x, y)
x = np.linspace(np.min(x)-100, np.max(x)+100, 1000)
y = np.array([lin(k) for k in x])
plt.plot(x, y, color='red', label='Regression Line')
plt.scatter(x, y, color='green', label='Scatter plot')
plt.xlabel('Head size (cm3)')
plt.ylabel('Brain Weight (grams)')
plt.legend()
plt.show()
```



Step 7: Plot the regression line:

\* Plot the original data points  $(X, y)$  as a scatter plot.

\* Plot the regression line  $(X, \text{predicted } y)$  as line plot.

Step 8: Calculate the R-squared value:

\* Calculate the total sum of squares  $TSS = \sum_{i=1}^n (y_i - \bar{y})^2$

\* Calculate the residual sum of squares (RSS) using the former

$$RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

\* R-squared value  $R^2 = \frac{RSS}{TSS}$

Step 9: Display the results:

\* Print the slope, intercept, & R-squared value.

Step 10: Complete the program:

\* Combine all the steps into a Python program.

```
(import numpy as np)
import pandas as pd
import matplotlib.pyplot as plt

# Load the data
data = pd.read_csv('data.csv')

# Extract features and target variable
X = data['X']
y = data['y']

# Calculate the mean of y
y_mean = y.mean()

# Calculate the total sum of squares (TSS)
TSS = ((y - y_mean)**2).sum()

# Calculate the slope and intercept
slope = ((X - X.mean()) * (y - y_mean)).sum() / ((X - X.mean())**2).sum()
intercept = y_mean - slope * X.mean()

# Calculate the predicted values
y_pred = intercept + slope * X

# Calculate the residual sum of squares (RSS)
RSS = ((y - y_pred)**2).sum()

# Calculate the R-squared value
R_squared = 1 - (RSS / TSS)

# Print the results
print("Slope: ", slope)
print("Intercept: ", intercept)
print("R-squared value: ", R_squared)
```

Result:

The Python program to implement simple linear regression using least square method for the given head brain dataset is analyzed and the linear regression line is constructed successfully.

(0.05 as 16, 100)



def get\_err(line = f'u0 in y):

y-m = np.mean(y)

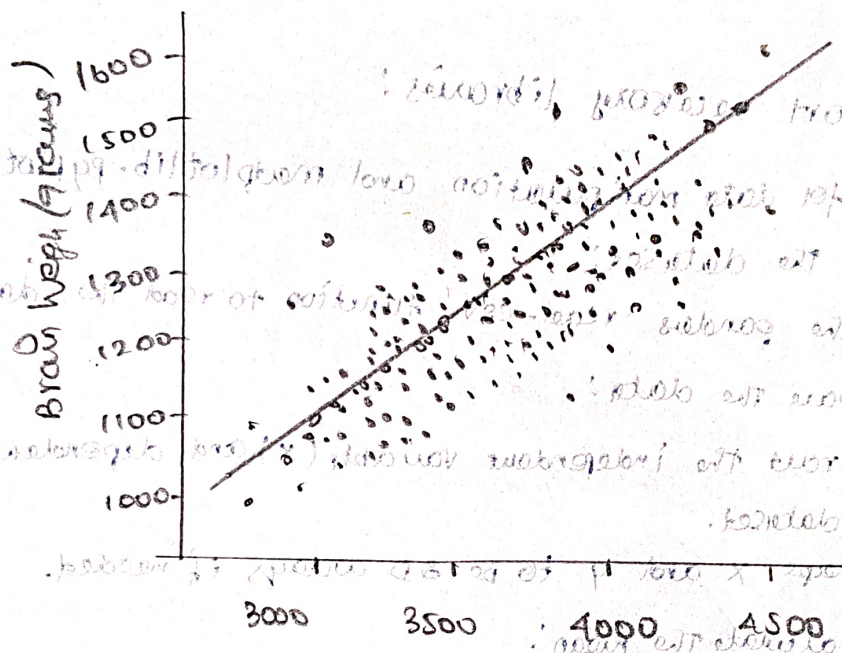
y-pred = np.array([line - f'u0 in x])

ss-t = np.sum((y - y-m)\*\*2)

ss-r = np.sum((y - y-pred)\*\*2)

return 1 - (ss-r/ss-t)

get\_err(line, x, y)



y = head size (cm³)

calculate the coefficient

calculate the slope (m) using the formula

$$m = \frac{\sum_{i=1}^n (y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

calculate the intercept (b) using the formula

calculate the intercept

use the calculated slope and intercept to write

equation for line of best fit

$$y = mx + b$$