Ex No: 2

A PYTHON PROGRAM TO IMPLEMENT SIMPLE LINEAR REGRESSION USING THE LEAST SQUARE METHOD

Aim:

To implement a Python program for constructing a simple linear regression using the 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 (e.g., headbrain.csv).

● Store the dataset in a variable (e.g., `data`).

Step 3: Prepare the data:

● Extract the independent variable (X) and dependent variable (y) 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 the formula:

$$b = \bar{y} - m\bar{X}$$

Step 6: Make predictions:

● Use the calculated slope and intercept to make predictions for each X value:

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

Step 7: Plot the regression line:

● Plot the original data points (X, y) as a scatter plot.

● Plot the regression line (X, predicted_y) as a line plot.

Step 8: Calculate the R-squared value:

● Calculate the total sum of squares (TSS) using the formula:

$$TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

● Calculate the residual sum of squares (RSS) using the formula:

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

● Calculate the R-squared value using the formula:

$$R^2 = 1 - \frac{RSS}{TSS}$$

Step 9: Display the results:

● Print the slope, intercept, and R-squared value.

Step 10: Complete the program:

● Combine all the steps into a Python program.

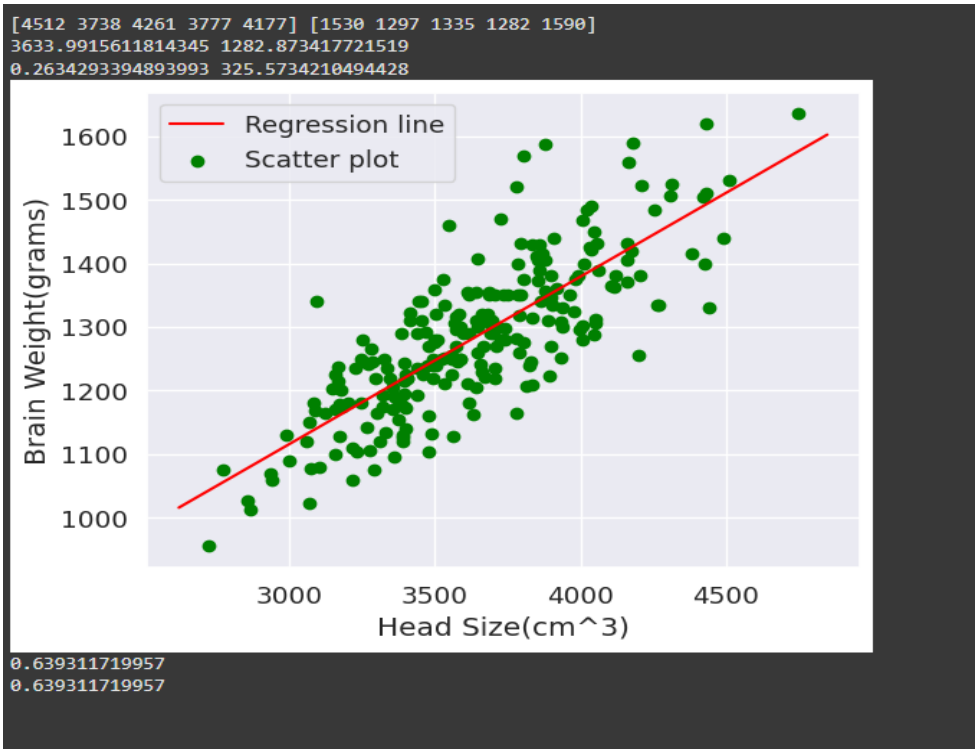
● Run the program to perform simple linear regression on the dataset.

PROGRAM:

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

data = pd.read_csv('/content/headbrain - headbrain.csv')
```

```
x, y = np.array(list(data['Head Size(cm^3)'])), np.array(list(data['Brain
Weight(grams)']))
print(x[:5], y[:5])

def get_line(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 x: m * x + c

lin = get_line(x, y)
X = np.linspace(np.min(x) - 100, np.max(x) + 100, 1000)
Y = np.array([lin(x) for x in X])

plt.plot(X, Y, color='red', label='Regression line')
plt.scatter(x, y, color='green', label='Scatter plot')
plt.xlabel('Head Size(cm^3)')
plt.ylabel('Brain Weight(grams)')
plt.legend()
plt.show()

def get_error(line_func, x, y):
    y_m = np.mean(y)
    y_pred = np.array([line_func(_) for _ in x])
    ss_t = np.sum((y - y_m) ** 2)
    ss_r = np.sum((y - y_pred) ** 2)
    return 1 - (ss_r / ss_t)

print(get_error(lin, x, y))

from sklearn.linear_model import LinearRegression
x = x.reshape((len(x), 1))
reg = LinearRegression()
reg = reg.fit(x, y)
print(reg.score(x, y))
```

OUTPUT:



RESULT:

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