Practical 6

- Shivam Tawari A-58

Aim: Write a program to implement linear regression in python.

Theory:

Linear Regression:

Linear regression is probably one of the most important and widely used regression techniques. It's among the simplest regression methods.

Single-variate linear regression is the simplest case of linear regression with a single independent variable.

Multi-variate linear regression is a case of linear regression with two or more independent variables.

The package scikit-learn is a widely used Python library for machine learning, built on top of Numpy and some other packages. It provides the means for preprocessing data, reducing dimensionality, implementing regression, classification, clustering, and more.

Steps :

① Step 1 : Import packages and classes

The first thing is to import the numey and LinearRegression from sklearn.linear_model. The class sklear.linear_model.LinearRegression will be used to perform linear and polynomial regression and make predictions.

② Step 2 : Provide Data

The second step is defining data to work with. The inputs (regressors, x) and output (predictor, y) shoud be arrays. x has two dimensions, while y has a single dimension.

③ Step 3 : Create a model and fit it

The next step is to create a linear regression model and fit it using the existing data.

model = LinearRegression ()

With .fit(), we calculate the optimal values of the weights b_0 and b_1, using the existing input and output (x and y) as the arguments.

④ Step 4 : Get Results

We can obtain coefficient of determination (R^2) with .score() called on model.

⑤ Step 5 : Predict Response

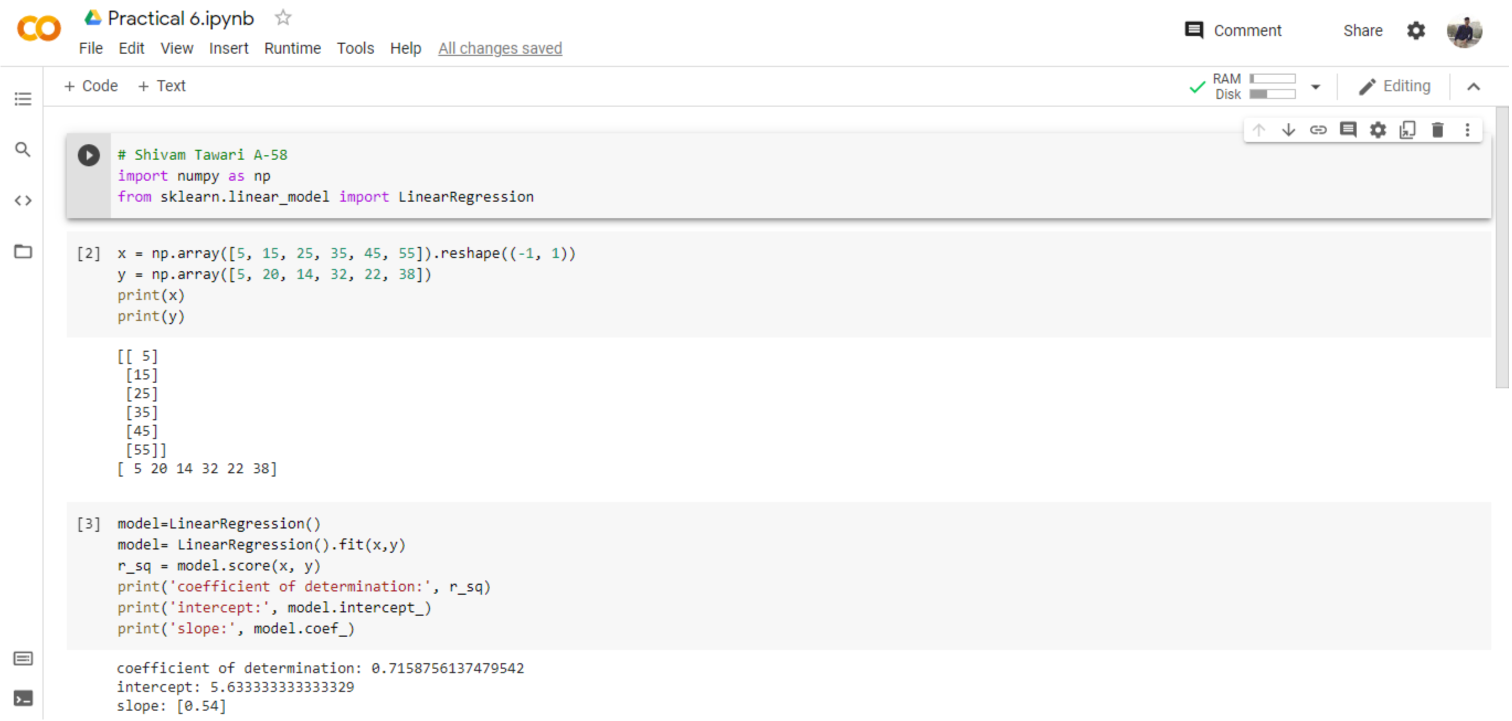
To obtain the predicted response, we use .predict().

Code:

```
import numpy as np
from sklearn.linear_model import LinearRegression

x = np.array([5,15,25,35,45,55]).reshape((-1,1))
y = np.array([5,20,14,32,22,38])
print (x)
print (y)

model = LinearRegression()
model = LinearRegression().fit(x,y)
r_sq = model.score(x,y)
print ('coefficient of determination:', r_sq)
print ('intercept :' model.intercept_)
print ('slope :', model.coef_)

x_pred = np.array([62]).reshape(-1,1)
y_pred = model.predict(x_predict)
print (y_pred)
```

Conclusion: We have successfully implemented a python program to perform Linear Regression.

Code & Output:



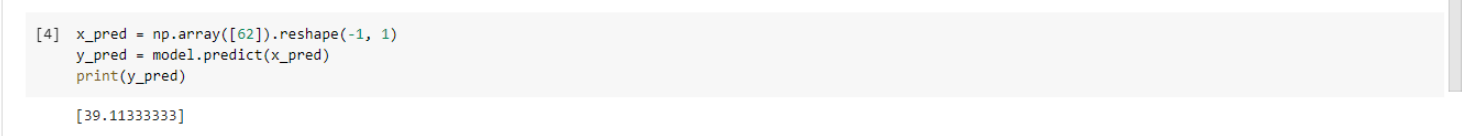
```
# Shivam Tawari A-58
import numpy as np
from sklearn.linear_model import LinearRegression
```

```
[2] x = np.array([5, 15, 25, 35, 45, 55]).reshape((-1, 1))
    y = np.array([5, 20, 14, 32, 22, 38])
    print(x)
    print(y)
```

```
[[ 5]
 [15]
 [25]
 [35]
 [45]
 [55]]
[ 5 20 14 32 22 38]
```

```
[3] model=LinearRegression()
    model= LinearRegression().fit(x,y)
    r_sq = model.score(x, y)
    print('coefficient of determination:', r_sq)
    print('intercept:', model.intercept_)
    print('slope:', model.coef_)
```

```
coefficient of determination: 0.7158756137479542
intercept: 5.633333333333329
slope: [0.54]
```



```
[4] x_pred = np.array([62]).reshape(-1, 1)
    y_pred = model.predict(x_pred)
    print(y_pred)
```

```
[39.11333333]
```