**LAB EXERCISE – 9**

**LINEAR REGRESSION AND MULTIPLE REGRESSION**

**Aim of the Experiment**

To write Python program for finding linear regression.

**Reference to Textbook and Explanation**

Chapter 5 and Appendix 2 for details about Linear and multiple Regression.

Consider the dataset given in Chapter 5 of this textbook. It is reproduced here for understanding in the Table below.

First one can write a linear regression for this problem and can verify that the results are same.

**Table 5.1:** SampleData

|  |  |
| --- | --- |
| **x**  **(Week)** | **y**  **(Sales in Thousands)** |
| 1 | 1.2 |
| 2 | 1.8 |
| 3 | 2.6 |
| 4 | 3.2 |
| 5 | 3.8 |

In listing 2, A random dataset is taken, and multiple regression is applied. This experiment will help to understand the concepts of multiple regression.

The command

X,y = make\_regression(n\_samples = 50,n\_features=1,noise=0.1)

Can create a regression dataset with 50 samples and 1 feature. The number of features field can be changed with 2 for multiple regression as

X,y = make\_regression(n\_samples = 50,n\_features=2,noise=0.1)

**WARNING – Random dataset is used for Listing 2 and 3. So, the dataset would be generated at every run. As dataset is generated again, the results would vary every time the program is run.**

**Listing - 1**

import matplotlib.pyplot as plt

import pandas as pd

from sklearn.linear\_model import LinearRegression

from sklearn import linear\_model

salesdata = {'week': [1,2,3,4,5],

'sales': [1.2,1.8,2.6,3.2,3.8]

}

df = pd.DataFrame(salesdata,columns=['week','sales'])

plt.scatter(df['week'], df['sales'], color='green')

plt.title('Regression among week and sales')

plt.xlabel('X - axis - Week')

plt.ylabel('Y- Dependent - Sales')

"""

week = df['week'].values.reshape(1,-1)

sales = df['sales'].values.reshape(1,-1)

"""

X = df[['week']]

y = df['sales']

regr = linear\_model.LinearRegression()

regr.fit(X,y)

print('Intercept: \n', regr.intercept\_)

print('Coefficients: \n', regr.coef\_)

print('\nThe Regression Equation is',regr.coef\_,'\* X+',regr.intercept\_)

# Fit the model for the given data

pred = regr.predict(X)

plt.plot(X,pred)

# Compute Adjusted R squared Error

print("\nAdjusted R Squared for Regression model:",regr.score(X,y))

**Output**

**Chart, line chart

Description automatically generated**

**Listing 2**

**WARNING – Random dataset is used for Listing 2. So, the random dataset would be generated at every run. As dataset is generated again, the results would vary every time the program is run.**

import matplotlib.pyplot as plt

from sklearn import linear\_model

from sklearn.datasets import make\_regression

X,y = make\_regression(n\_samples = 50,n\_features=1,noise=0.1)

plt.scatter(X,y,color='green')

plt.title('Regression among X and y')

plt.xlabel('X - axis - X')

plt.ylabel('Y- Dependent - y')

regr = linear\_model.LinearRegression()

regr.fit(X,y)

print('Intercept: \n', regr.intercept\_)

print('Coefficients: \n', regr.coef\_)

print('\nThe Regression Equation is',regr.coef\_,'\* X +',regr.intercept\_)

# Fit the model for the given data

pred = regr.predict(X)

plt.plot(X,pred)

# Compute Adjusted R squared Error

print("\nAdjusted R Squared for Regression model:",regr.score(X,y))

**Output**

Chart, line chart

Description automatically generated

**Listing 3**

**WARNING – Random dataset is used for Listing 3. So, the random dataset would be generated at every run. As dataset is generated again, the results would vary every time the program is run.**

**Multiple Regression**

from sklearn import linear\_model

from sklearn.datasets import make\_regression

print("Multiple regression \n\n")

# Multiple Regression

# Create random dataset with 2 features. Dataset has 50 samples with noise 0.1.

X,y = make\_regression(n\_samples = 50,n\_features=2,noise=0.1)

regr = linear\_model.LinearRegression()

regr.fit(X,y)

print('Intercept: \n', regr.intercept\_)

print('Coefficients: \n', regr.coef\_)

# Compute Adjusted R squared Error

print("\nAdjusted R Squared for Regression model:",regr.score(X,y))

**Output**

Graphical user interface

Description automatically generated