The Spark Foundation Data Science and Business Analytics internship

Task 1:Linear regression model for predicting scores of students

Author:Tom Roy Thomas

In this exercise we will be using the python code to predict the student score by taking the hours they study per day as unit. Using the Machine Learning technique called simple liner regression we will build a model

Step 1:Importing all neccessary libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import linear_model
```

Step 2:Load Data

```
data="http://bit.ly/w-data"
df=pd.read_csv(data)
print(df)
```

```
    Hours  Scores
0     2.5      21
1     5.1      47
2     3.2      27
3     8.5      75
4     3.5      30
5     1.5      20
6     9.2      88
7     5.5      60
8     8.3      81
9     2.7      25
10    7.7      85
11    5.9      62
12    4.5      41
13    3.3      42
14    1.1      17
15    8.9      95
16    2.5      30
17    1.9      24
18    6.1      67
19    7.4      69
20    2.7      30
21    4.8      54
22    3.8      35
23    6.9      76
24    7.8      86
```

```
df.shape
```

```
(25, 2)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 528.0 bytes
```

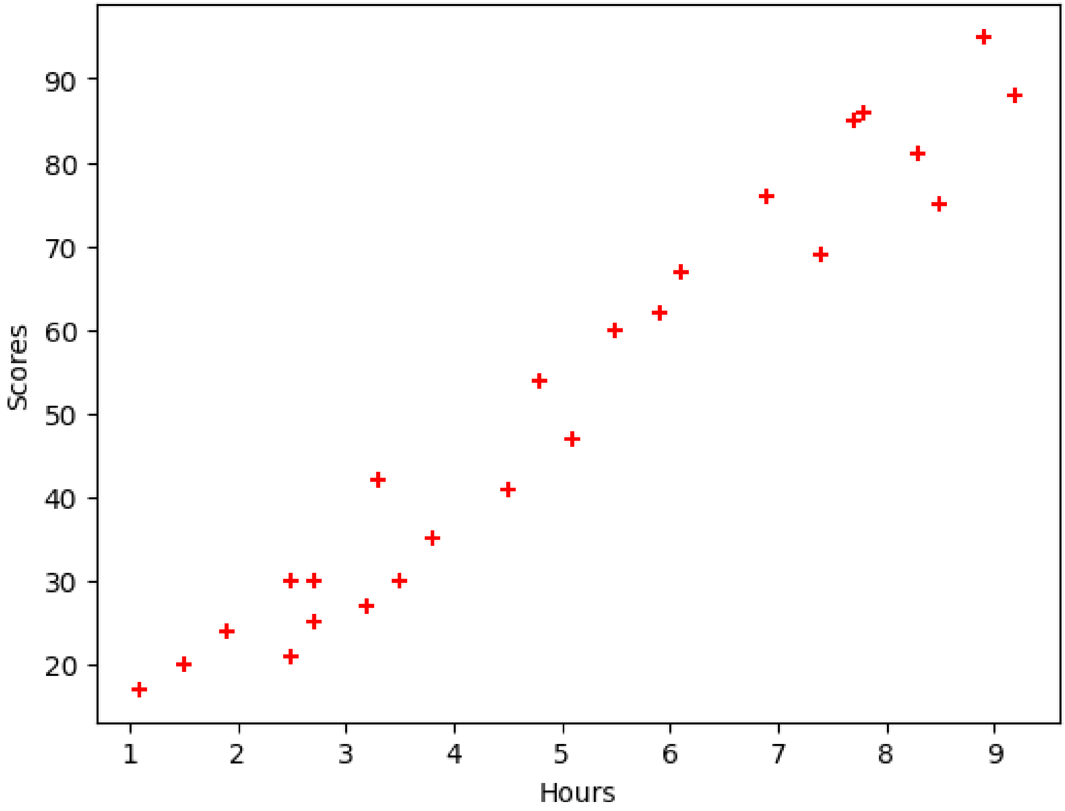
```
df.describe()
```

```
           Hours     Scores
count  25.000000  25.000000
mean    5.012000  51.480000
std     2.525094  25.286887
min     1.100000  17.000000
25%     2.700000  30.000000
50%     4.800000  47.000000
75%     7.400000  75.000000
max     9.200000  95.000000
```

Step 3: Data Visualization

```
plt.scatter(df.Hours,df.Scores,color="r",marker="+")
plt.xlabel("Hours")
plt.ylabel("Scores")
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



Step 4:Dataset Training and Model Building

Here we are using linear regression from scikit.learn library to create the model.After creating model we will train the model using dataset.

```
x=df.drop("Scores",axis='columns')
x
```

```
    Hours
0     2.5
1     5.1
2     3.2
3     8.5
4     3.5
5     1.5
6     9.2
7     5.5
8     8.3
9     2.7
10    7.7
11    5.9
12    4.5
13    3.3
14    1.1
15    8.9
```

```
16    2.5
17    1.9
18    6.1
19    7.4
20    2.7
21    4.8
22    3.8
23    6.9
24    7.8
```

```
y=df['Scores']
y
```

```
0     21
1     47
2     27
3     75
4     30
5     20
6     88
7     60
8     81
9     25
10    85
11    62
12    41
13    42
14    17
15    95
16    30
17    24
18    67
19    69
20    30
21    54
22    35
23    76
24    86
Name: Scores, dtype: int64
```

```
model=linear_model.LinearRegression()
model
```

```
LinearRegression()
```

```
model.fit(x,y)
```

```
LinearRegression()
```

Step 5:Model Evaluation

```
model.score( df[['Hours']],df[['Scores']])
```

```
0.9529481969048356
```

The score of the model indicates that it has an accuracy of around 95% and it is a good score.So while training ,model will try to find the best.

```
model.coef_
```

```
array([9.77580339])
```

```
model.intercept_
```

```
2.48367340537321
```

What will be predicted score if a student studies for 9.25hrs/day?

```
model.predict([[9.25]])
```

```
c:\Users\USER01\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:439: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

```
array([92.90985477])
```

```
x=9.7758*9.25+2.4836
x
```

```
92.90975
```

It is found that predicted score by the model and by giving direct substituton is almost equal.

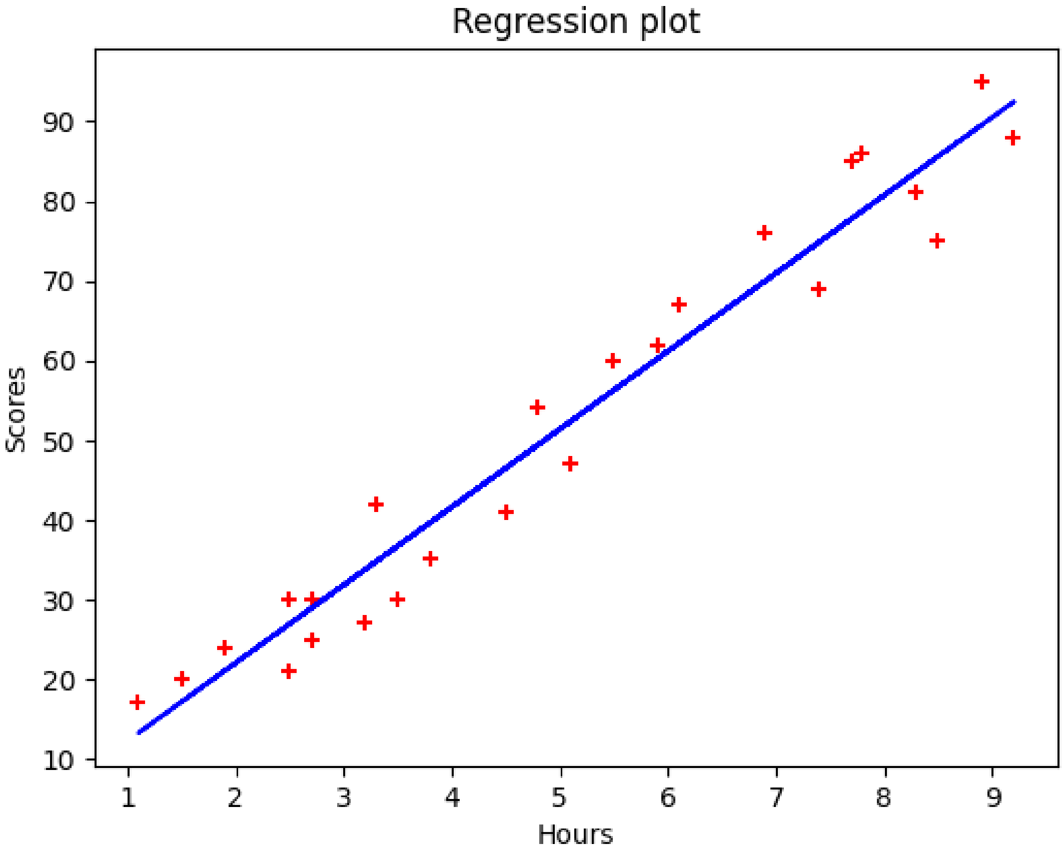
```
df['predicted_scores']=model.predict(df[['Hours']])
df
```

```
    Hours  Scores  predicted_scores
0     2.5      21         26.923182
1     5.1      47         52.340271
2     3.2      27         33.766244
3     8.5      75         85.578002
4     3.5      30         36.698985
5     1.5      20         17.147378
6     9.2      88         92.421065
7     5.5      60         56.250592
8     8.3      81         83.622842
9     2.7      25         28.878343
10    7.7      85         77.757360
11    5.9      62         60.160913
12    4.5      41         46.474789
13    3.3      42         34.743825
14    1.1      17         13.237057
15    8.9      95         89.488324
16    2.5      30         26.923182
```

```
17     1.9      24          21.057700
18     6.1      67          62.116074
19     7.4      69          74.824618
20     2.7      30          28.878343
21     4.8      54          49.407530
22     3.8      35          39.631726
23     6.9      76          69.936717
24     7.8      86          78.734940
```

Let us plot graph to see how well is the model performing

```
plt.scatter(df.Hours,df.Scores,color="r",marker="+")
plt.plot(df.Hours,df["predicted_scores"],color='b')
plt.xlabel("Hours")
plt.ylabel("Scores")
plt.title("Regression plot")
plt.show()
```


Regression plot
Scores
Hours