Prediction Using Supervised ML

In [44]:
```
#In this task we are going to predict the percentage of marks that a student is expected to score.
#Based upon the number hours they studied.
#This is a simple linear regression task in which only two variables.
#We are going step by step:
```

Author: Rudra Pratap Singh

In [2]:
```
#If we have not any library which is necessary to perform the task then we have to install it.
!pip install sklearn
```

```
Requirement already satisfied: sklearn in c:\users\rudra\appdata\local\programs\python\python38-32\lib\site-packages (0.0)
WARNING: You are using pip version 20.2.4; however, version 21.1.3 is available.
You should consider upgrading via the 'c:\users\rudra\appdata\local\programs\python\python38-32\python.exe -m pip install --upgrade pip' command.
Requirement already satisfied: scikit-learn in c:\users\rudra\appdata\local\programs\python\python38-32\lib\site-packages (from sklearn) (0.24.2)
Requirement already satisfied: scipy>=0.19.1 in c:\users\rudra\appdata\local\programs\python\python38-32\lib\site-packages (from scikit-learn->skle
arn) (1.7.0)
Requirement already satisfied: numpy>=1.13.3 in c:\users\rudra\appdata\local\programs\python\python38-32\lib\site-packages (from scikit-learn->skle
arn) (1.20.3)
Requirement already satisfied: joblib>=0.11 in c:\users\rudra\appdata\local\programs\python\python38-32\lib\site-packages (from scikit-learn->sklea
rn) (1.0.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\rudra\appdata\local\programs\python\python38-32\lib\site-packages (from scikit-lear
n->sklearn) (2.1.0)
```

Importing Libraries

In [6]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split as tts
```

Reading Data

In [7]:
```
url='http://bit.ly/w-data'
print("Your Given Data is:")
file=pd.read_csv(url)
file.head(25)
```

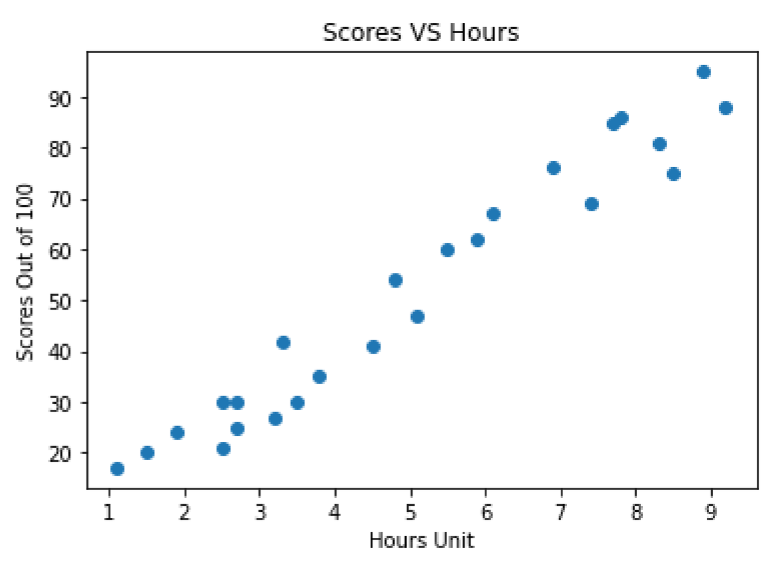
Your Given Data is:

Out[7]:

	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

Scatter Plot Scores VS Hours

In [8]:
```
plt.scatter(x=file.Hours,y=file.Scores)
plt.xlabel("Hours Unit")
plt.ylabel("Scores Out of 100")
plt.title("Scores VS Hours")
plt.show()
```



In [27]:

In [9]:
```
file.describe()
```

Out[9]:

	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

In [10]:
```
file.shape
```

Out[10]: (25, 2)

In [11]:
```
file.columns
```

Out[11]: Index(['Hours', 'Scores'], dtype='object')

Preparing and Visualization of Data

In [12]:
```
#Sklearn train_test_split will make random partitions for the two subsets.
#However, you can also specify a random state for the operation.
#train_test_split(X, y, train_size=0.*,test_size=0.*, random_state=*)
trn,tst=tts(file,train_size=0.20,random_state=113)
```

In [13]:
```
trn.shape
```

Out[13]: (5, 2)

In [14]:
```
tst.shape
```

Out[14]: (20, 2)

In [15]:
```
tst
```

Out[15]:

	Hours	Scores
18	6.1	67
19	7.4	69
3	8.5	75
8	8.3	81
13	3.3	42
14	1.1	17
1	5.1	47
6	9.2	88
9	2.7	25
24	7.8	86
4	3.5	30
17	1.9	24
11	5.9	62
7	5.5	60
22	3.8	35
21	4.8	54
20	2.7	30
15	8.9	95
0	2.5	21
16	2.5	30

In [16]:
```
trn
```

Out[16]:

	Hours	Scores
23	6.9	76
12	4.5	41
10	7.7	85
2	3.2	27
5	1.5	20

In [17]:
```
trn_x=trn.drop('Scores',axis=1)
trn_y=trn['Scores']
```

In [18]:
```
tst_x=tst.drop('Scores',axis=1)
tst_y=tst['Scores']
```

Implimenting The Algorithm

In [19]:
```
lr=LinearRegression()
```

In [20]:
```
lr.fit(trn_x,trn_y)
```

Out[20]: LinearRegression()

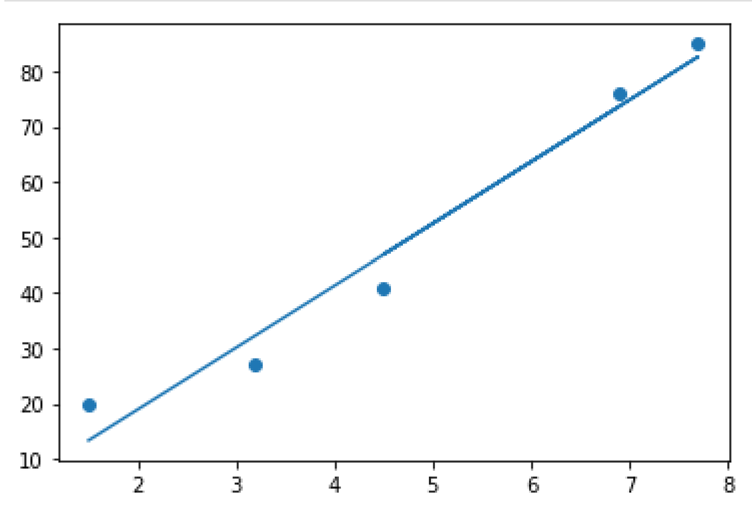
In [21]:
```
lr.intercept_
```

Out[21]: -3.406800242865806

In [22]:
```
lr.coef_
```

Out[22]: array([11.17789921])

In [23]:
```
#we have to plot the regression line using formula y=m*x+c vs Ploting the file data which is trn_x and trn_y
ln=lr.coef_*trn_x+lr.intercept_
plt.plot(trn_x,ln)# this should be a straight line
plt.scatter(trn_x,trn_y)
plt.show()
```



Predictions of data

In [43]:
```
#Predict() function takes 2 dimensional array as arguments.So,If we want to predict the value for simple linear regression.
#Then we have to issue the prediction value within 2 dimentional array.
prediction=lr.predict(tst_x)
```

In [26]:
```
final=list(zip(tst_y,prediction))
```

In [27]:
```
final
```

Out[27]:
```
[(67, 64.77838494231938),
 (69, 79.30965391621129),
 (75, 91.60534304796599),
 (81, 89.36976320582878),
 (42, 33.480267152398305),
 (17, 8.888888888888898),
 (47, 53.60048573163327),
 (88, 99.42987249544626),
 (25, 26.773527625986652),
 (86, 83.78081360048571),
 (30, 35.715846994535525),
 (24, 17.831208257437773),
 (62, 62.54280510018216),
 (60, 58.07164541590771),
 (35, 39.06921675774135),
 (54, 50.247115968427444),
 (30, 26.773527625986652),
 (95, 96.07650273224044),
 (21, 24.53794778384943),
 (30, 24.53794778384943)]
```

Finally we have to evaluate our model

In [30]:
```
#for finding mean square error we have to import mean_squared_error from sklearn.matrics
from sklearn.metrics import mean_squared_error as mse
```

In [36]:
```
q=mse(tst_y,prediction,squared=False)
```

In [37]:
```
q
```

Out[37]: 6.844597394845228

Solution for the given task

In [40]:
```
#we have to find predicted score if a student studies for 9.25 hrs/ day?
hrs=[9.25]
new_prediction=lr.predict([hrs])
```

In [41]:
```
#we have to print the predicted score
print("Predicted Score:",new_prediction)
```

Predicted Score: [99.98876746]

Predicted Score: [99.98876746]