The Sparks Foundation - GRIP - Data Science and Business Analytics Intern - JULY-2021

TASK 1 - Prediction using Supervised ML

by KARTHIK SUNKARI

In this task We are going Predicting the percentage of an student based on the number of study hours using linear regression algorithm

Step1 Defining objectives

In [4]:

```
#importing nessessary libraries
import sklearn
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sn
```

Step2 Data collection

In [5]:
```
#importing the dataset and displaying
dt=pd.read_csv("http://bit.ly/w-data")
dt
```

Out[5]:

	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

Step3 Data Preprocessing

In [6]:

```
dt.describe()
```

Out[6]:

	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 [7]:

```
#checking the null values
dt.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
```

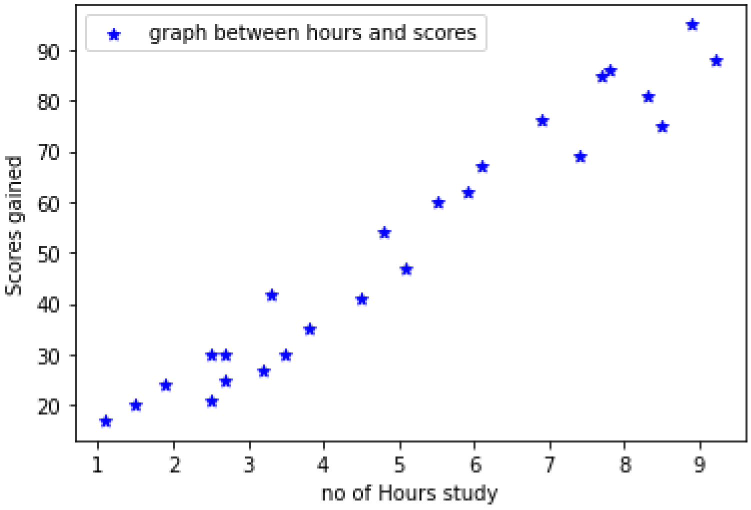
Step4 Data Visualization

In [8]:

```
x=dt["Hours"]
y=dt["Scores"]
plt.xlabel("no of Hours study")
plt.ylabel("Scores gained")
plt.scatter(x,y,marker="*",color="blue",label="graph between hours and scores")
plt.legend()
```

Out[8]:

```
<matplotlib.legend.Legend at 0x1f94d35e220>
```



Step5 Spliting of dataset into testing and training/Model selection

In [9]:

```
x=dt.iloc[:,:-1].values
y=dt.iloc[:,-1].values

from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=1/3,random_state=1)
```

Creating simple linear model

In [10]:

```
from sklearn.linear_model import LinearRegression
model=LinearRegression()           # which creates linear equation y=ax+b
model.fit(xtrain,ytrain)
```

Out[10]:

```
LinearRegression()
```

Step6 Prediction of data/ Model Building

In [11]:

```
y_pred=model.predict(xtest)
y_pred
```

Out[11]:

```
array([10.56351243, 33.29165695, 18.82829225, 87.01272581, 48.78811912,
       78.74794599, 62.21838634, 75.64865355, 35.3578519 ])
```

comparing actual vs predicted

In [12]:
```
dt=pd.DataFrame({'Actual':ytest,'Predicted':y_pred})
dt
```

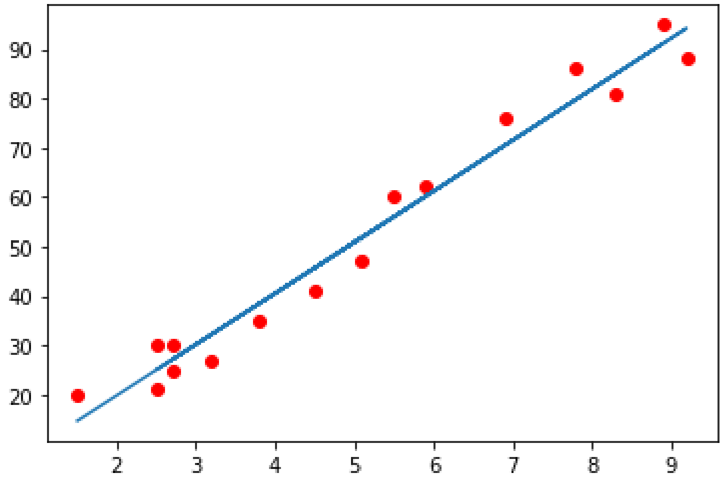
Out[12]:

	Actual	Predicted
0	17	10.563512
1	42	33.291657
2	24	18.828292
3	75	87.012726
4	54	48.788119
5	85	78.747946
6	67	62.218386
7	69	75.648654
8	30	35.357852

checking training data prediction

In [13]:
```
plt.scatter(xtrain,ytrain,color="red")
plt.plot(xtrain,model.predict(xtrain))
```

Out[13]:
```
[<matplotlib.lines.Line2D at 0x1f94db603d0>]
```



In [14]:
```
print("model cofficient",model.coef_)
print("model interception",model.intercept_)
```

```
model cofficient [10.33097478]
model interception -0.8005598320504035
```

In [15]:
```
print("Training Accuracy:",model.score(xtrain,ytrain),"\nTesting Accuracy:",mode
```

```
Training Accuracy: 0.9693800724956538
Testing Accuracy: 0.9047140370739192
```

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

In [16]:

```
hours=9.25
pred=model.predict([[hours]])
print(f"student studies for {hours} his estimated score will be {float(pred)}")
```

```
student studies for 9.25 his estimated score will be 94.76095689811578
```

In [ ]: