The Spark Foundation:

Data Science and Business Analytics Intern- Feb'22

Name:- Akanksha Shivaji Nangare

In given task we have to predict the percentage of marks excepted by the student based on the number of hours they studied. In this task only two variables are involved.

In [1]:

```
#Importing all libraries required in tis notebook
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
get_ipython().run_line_magic('matplotlib', 'inline')
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
```

In [2]:

```
#Importing data
url="http://bit.ly/w-data"
data=pd.read_csv(url)
data1=data
print("The data is imported successfully")
data
```

```
The data is imported successfully
```

Out[2]:

	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

In [3]:

```
data.describe()
```

Out[3]:

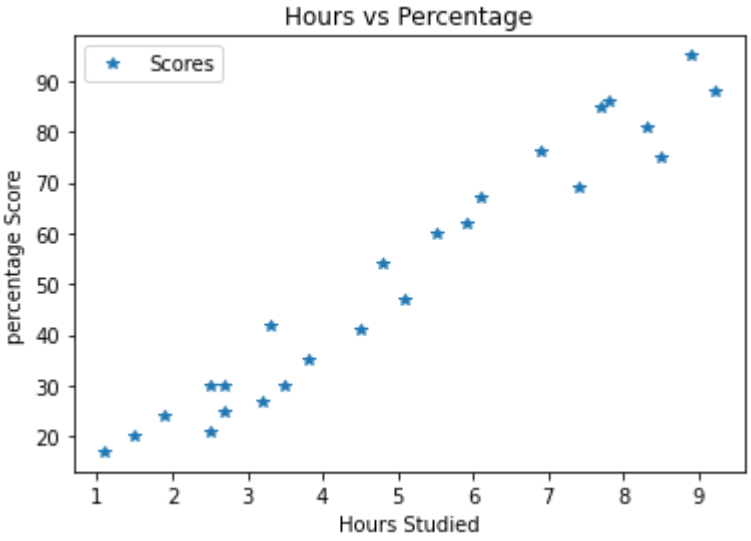
	Hours	Scores
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

Data visualization

Now let's plot a graph of given data so that it will ive us clear idea about data.

In [6]:

```
#plotting the distribution of scores
data.plot(x='Hours',y='Scores',style='*')
plt.title('Hours vs Percentage')
plt.xlabel('Hours Studied')
plt.ylabel('percentage Score')
plt.show()
```



Linear Regression Model

Now we prepare the data and split in test data

In [8]:

```
#splitting trainin and testing data
x=data.iloc[:,:-1].values
y=data.iloc[:,1].values
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.80,test_size=0.20,random_st
```

Training the Model

In [10]:

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
print("Train the Model")
#y_predict=lr.predict(x_train)
```

```
Train the Model
```

Training the Algorithm

Now the spliting of our data into training and testing set is done , now it's time to train our algorithm.

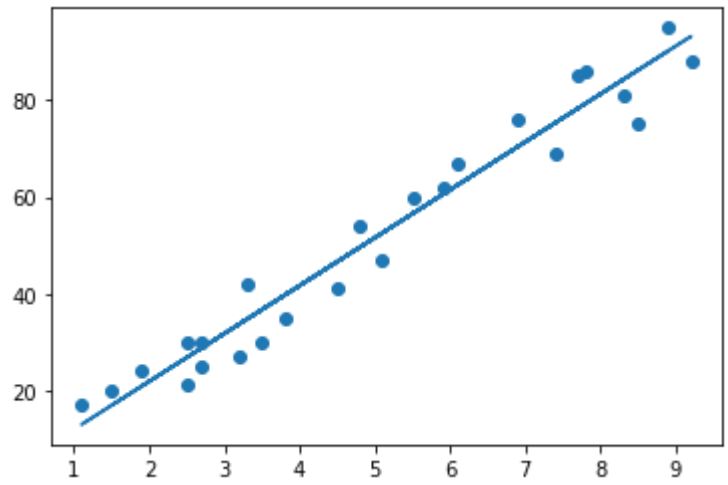
In [11]:

```
lr=LinearRegression()
lr.fit(x_train,y_train)

print("Training Completed.")
```

```
Training Completed.
```

In [12]:

```
#plotting the regression line
l=lr.coef_*x+lr.intercept_

#plotting for the test data
plt.scatter(x,y)
plt.plot(x,l);
plt.show()
```



Checking the accuracy scores for training and test set

In [13]:

```
print('Test Score')
print(lr.score(x_test,y_test))
print('Training Score')
print(lr.score(x_train,y_train))
```

```
Test Score
0.9454906892105356
Training Score
0.9515510725211552
```

Now we make predictions

In [17]:

```
#testing data in Hours
print(x_test)
```

```
[[1.5]
 [3.2]
 [7.4]
 [2.5]
 [5.9]]
```

In [18]:

```
#predicting the scrore
y_pred=lr.predict(x_test)
print(y_pred)
```

```
[16.88414476 33.73226078 75.357018   26.79480124 60.49103328]
```

In [19]:

```
# Compairing actual vs predicted
final = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
final
```

Out[19]:

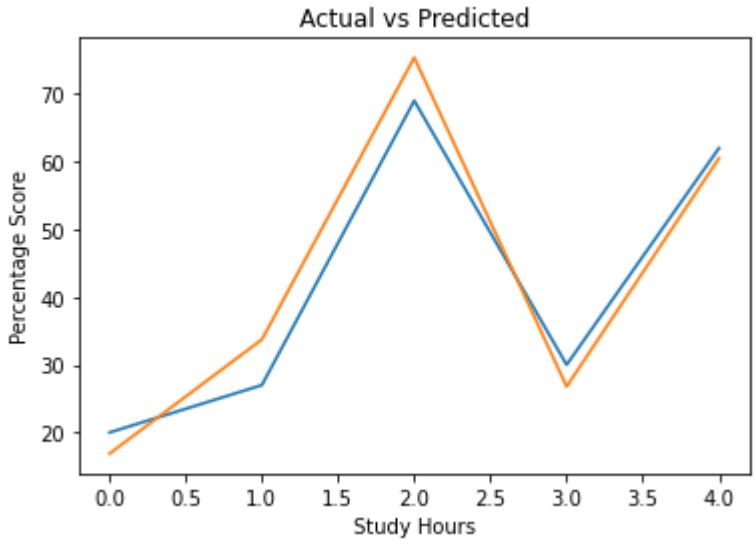
	Actual	Predicted
0	20	16.884145
1	27	33.732261
2	69	75.357018
3	30	26.794801
4	62	60.491033

In [20]:

```
plt.plot(final)
plt.xlabel('Study Hours')
plt.ylabel('Percentage Score')
plt.title('Actual vs Predicted')
```

Out[20]:

```
Text(0.5, 1.0, 'Actual vs Predicted')
```



Score prediction for 9.25 hours per day study.

In [22]:

```
predict_value=lr.predict([[9.00]])
print('Score prection if student study 9.25 hr/day is :')
predict_value[0]
```

```
Score prection if student study 9.25 hr/day is :
```

Out[22]:

```
91.21406836721481
```

Model Evaluation Metrics

In [23]:

```
#cheacking the efficiency of model
mean_squ_error = mean_squared_error(y_test, y_pred)
mean_abs_error = mean_absolute_error(y_test, y_pred)
print("Mean Squared Error :",mean_squ_error)
print("Mean Absolute Error :",mean_abs_error)
```

```
Mean Squared Error : 21.5987693072174
Mean Absolute Error : 4.183859899002975
```

Thus, with this, our task1 of PREDICTION USING SUPERVISED ML is Completed.

Thank you!

In [ ]:

```

```

In [ ]:

```

```