TASK 1 - Prediction using Supervised ML

To Predict the percentage of marks of the students based on the number of hours they studied

Author - VISHAL PANDEY

In [1]:
```
# imported the libraries
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import mean_absolute_error
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

In [2]:
```
#reading the data
dataset=pd.read_csv("http://bit.ly/w-data")
print('Data is sucessfully imported')
dataset
```

```
Data is sucessfully imported
```

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

	Hours	Scores
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

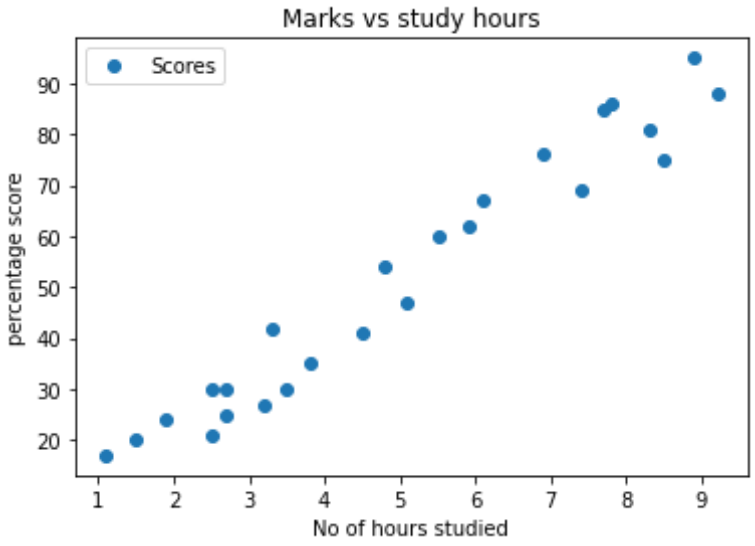
In [4]:
```
# Check if there any null value in the Dataset
dataset.isnull == True
```

Out[4]: False

Now Visualizing our Data

In [15]:
```
#plotting the distrubution of scores.
dataset.plot(x='Hours',y='Scores',style='o')
plt.title('Marks vs study hours')
```

```
plt.xlabel('No of hours studied')
plt.ylabel('percentage score')
plt.show()
```



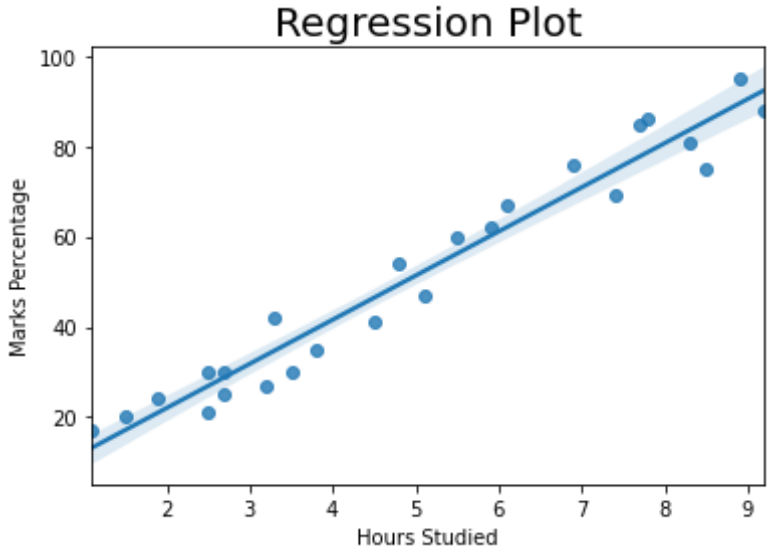
In [18]:
```
dataset.describe()
```

Out[18]:

	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

Lets plot a regression line to confirm the correlation.

In [11]:
```
sns.regplot(x= dataset['Hours'], y= dataset['Scores'])
plt.title('Regression Plot',size=20)
plt.ylabel('Marks Percentage', size=10)
plt.xlabel('Hours Studied', size=10)
plt.show()
print(dataset.corr())
```




```
            Hours    Scores
Hours    1.000000  0.976191
Scores   0.976191  1.000000
```

It is confirmed that the variables are positively correlated

Model Training

1) Data splitting

In [14]:
```
# Defining X and y from the Dataset.
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 1].values

# Spliting the Dataset in two.
train_X, val_X, train_y, val_y = train_test_split(X, y, random_state = 0)
```

2) Fitting the Data into the model

In [16]:
```
regression = LinearRegression()
regression.fit(train_X, train_y)
print("**Model has been trained**")
```

```
**Model has been trained**
```

Marks Percentage prediction

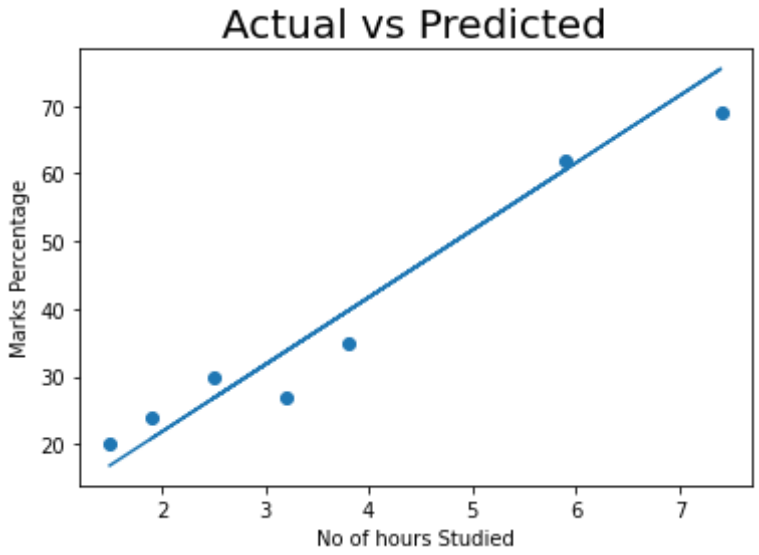
In [17]:
```
pred_y = regression.predict(val_X)
prediction = pd.DataFrame({'Hours': [i[0] for i in val_X], 'marks predicted': [k for k in pred_y]})
prediction
```

Out[17]:

	Hours	marks predicted
0	1.5	16.844722
1	3.2	33.745575
2	7.4	75.500624
3	2.5	26.786400
4	5.9	60.588106
5	3.8	39.710582
6	1.9	20.821393

Comparing the Predicted Marks with the Actual Marks

In [19]:
```
plt.scatter(x=val_X, y=val_y,)
plt.plot(val_X, pred_y,)
plt.title('Actual vs Predicted', size=20)
plt.ylabel('Marks Percentage', size=10)
plt.xlabel('No of hours Studied', size=10)
plt.show()
```



Evaluating the Model

In [20]:
```
print('Mean absolute error: ',mean_absolute_error(val_y,pred_y))
```

```
Mean absolute error:  4.130879918502486
```

Small value of Mean absolute error states that the chances of error or wrong forecasting through the model are very less.

What will be predicted Score if they studied 9.25hrs/day?

In [21]:
```
hours = [9.25]
answer = regression.predict([hours])
print("Score = {}".format(round(answer[0],4)))
```

```
Score = 93.8927
```

In [ ]:
```
```