Name: Ankita chaubey

Task 1 predict percentage based on learning hours using Linear Regression

In [3]:
```
#import all libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import r2_score,mean_squared_error
from math import sqrt

%matplotlib inline
```

In [4]:
```
data=pd.read_csv("C:/Users/ankit/Downloads/student_scores.csv")
```

Data Exploration:

In [5]:
```
data.head(10)
```

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

In [6]:
```
data.shape
```

Out[6]:
```
(25, 2)
```

In [7]:
```
data.describe()
```

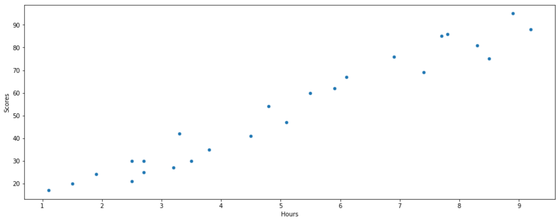
Out[7]:

	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

visualization

In [8]:
```
data.plot(kind="scatter",x="Hours",y="Scores",figsize=(16,6))
```

Out[8]:
```
<AxesSubplot:xlabel='Hours', ylabel='Scores'>
```



In [9]:
```
x = data.iloc[:,:-1].values
y = data.iloc[:,1].values
```

Training the Algorithm

In [10]:
```
from sklearn.model_selection import train_test_split

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 1/3, random_state = 0)
```

In [11]:
```
from sklearn.linear_model import LinearRegression


regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Out[11]:
```
LinearRegression()
```

Making Predictions

In [13]:
```
y_pred = regressor.predict(X_test)
print(y_pred)
```

```
[17.04289179 33.51695377 74.21757747 26.73351648 59.68164043 39.33132858
 20.91914167 78.09382734 69.37226512]
```

In [16]:
```
from statsmodels.sandbox.regression.predstd import wls_prediction_std
import statsmodels.api as sm
```

In [19]:
```
model1=sm.OLS(y_train,X_train)
result = model1.fit()
result.summary()
```

```
C:\Users\ankit\anaconda3\lib\site-packages\scipy\stats\stats.py:1603: UserWarning: kurtosistest only valid for n>=20 ... continuing anyway, n=16
  warnings.warn("kurtosistest only valid for n>=20 ... continuing "
```

Out[19]:

OLS Regression Results

Dep. Variable:	y	**R-squared (uncentered):**	0.991
Model:	OLS	**Adj. R-squared (uncentered):**	0.990
Method:	Least Squares	**F-statistic:**	1611.
Date:	Thu, 04 Feb 2021	**Prob (F-statistic):**	1.11e-16
Time:	09:49:43	**Log-Likelihood:**	-50.502
No. Observations:	16	**AIC:**	103.0
Df Residuals:	15	**BIC:**	103.8
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
x1	10.0780	0.251	40.132	0.000	9.543	10.613

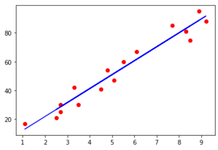
Omnibus:	2.476	**Durbin-Watson:**	2.079
Prob(Omnibus):	0.290	**Jarque-Bera (JB):**	1.124
Skew:	-0.191	**Prob(JB):**	0.570
Kurtosis:	1.759	**Cond. No.**	1.00

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Visualising the Training set results

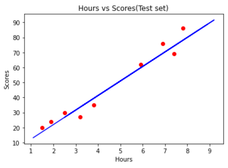
In [21]:
```
plt.scatter(X_train, y_train, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
```

Out[21]:
```
[<matplotlib.lines.Line2D at 0x20455bcb1c0>]
```



Visualising the Test set results

In [22]:
```
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Hours vs Scores(Test set)')
plt.xlabel('Hours')
plt.ylabel('Scores')
plt.show()
```



Question

what will be prdicted score if a student studies for 9.25 hrs/day?

In [23]:
```
Hr=pd.DataFrame({'Hours':[9.25]})
regressor.predict(Hr)
```

Out[23]:
```
array([92.14523315])
```

Evaluating the model

In [27]:
```
# from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_absolute_error
y_pre=regressor.predict(X_test)
mae = mean_absolute_error(y_test,y_pre)
```

In [28]:
```
mae
```

Out[28]:
```
4.691397441397438
```