AASTHA SINGH

Task 1: Data Science And Business Analytics Internship by The Sparks Foundation

Prediction Using Supervised ML

Prediction Percentage of Student based on number of study hours

In [1]:
```
#necessary Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
import statsmodels.api as sm

from sklearn.linear_model import LinearRegression
```

In [2]:
```
#Reading Data

data = pd.read_csv("Fbpio.csv")
data
```

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.head()
```

Out[3]:

	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30

In [4]:
```
data.tail()
```

Out[4]:

	Hours	Scores
20	2.7	30
21	4.8	54
22	3.8	35
23	6.9	76
24	7.8	86

To check the missing values

In [5]:
```
data.isnull().sum()
```

Out[5]:
```
Hours      0
Scores     0
dtype: int64
```

To generate descriptive statistics

In [6]:
```
data.describe()
```

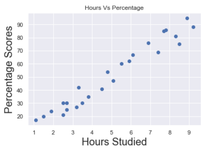
Out[6]:

	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

Data Visualization

In [10]:
```
x= data['Hours']
y= data['Scores']
```

In [11]:
```
plt.scatter(x,y)
plt.title('Hours Vs Percentage')
plt.xlabel('Hours Studied',fontsize=20)
plt.ylabel('Percentage Scores',fontsize=20)
plt.show()
```



Linear Regression Model

Preparing Data

In [12]:
```
x= data.iloc[:,:-1].values
y= data.iloc[:, 1].values
```

In [14]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_state=0)
```

In [15]:
```
regressor = LinearRegression()
regressor.fit(x_train,y_train)
print("Training Completed")
```

```
Training Completed
```

In [16]:
```
x1=sm.add_constant(x)
results = sm.OLS(y,x1).fit()
results.summary()
```

Out[16]:

OLS Regression Results

Dep. Variable:	y	R-squared:	0.953
Model:	OLS	Adj. R-squared:	0.951
Method:	Least Squares	F-statistic:	465.8
Date:	Wed, 19 May 2021	Prob (F-statistic):	9.13e-17
Time:	10:57:08	Log-Likelihood:	-77.514
No. Observations:	25	AIC:	159.0
Df Residuals:	23	BIC:	161.5
Df Model:	1		
Covariance Type:	nonrobust		

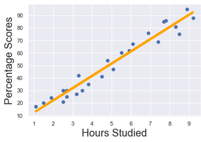
	coef	std err	t	P>|t|	[0.025	0.975]
const	2.4837	2.532	0.981	0.337	-2.753	7.721
x1	9.7758	0.453	21.583	0.000	8.839	10.713

Omnibus:	7.616	Durbin-Watson:	1.460
Prob(Omnibus):	0.022	Jarque-Bera (JB):	2.137
Skew:	-0.216	Prob(JB):	0.343
Kurtosis:	1.634	Cond. No.	13.0

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [18]:
```
#plotting the regression line

plt.scatter(x,y)
yhat= 9.7758*x+2.4837
fig= plt.plot(x,yhat,lw = 4,c ='orange',label = 'regression line')
plt.xlabel('Hours Studied',fontsize= 20)
plt.ylabel('Percentage Scores',fontsize= 20)
plt.show()
```



Marking Prediction

In [19]:
```
print(x_test)
y_pred = regressor.predict(x_test)
```

```
[[1.5]
 [3.2]
 [7.4]
 [2.5]
 [5.9]
 [3.8]
 [1.9]
 [7.8]]
```

In [20]:
```
#comparing actual vs predicted values

df=pd.DataFrame({'Actual': y_test, 'Predicted':y_pred})
df
```

Out[20]:

	Actual	Predicted
0	20	17.053665
1	27	33.694229
2	69	74.806209
3	30	26.842232
4	62	60.123359
5	35	39.567369
6	24	20.969092
7	86	78.721636

Evaluation Model

In [21]:
```
from sklearn import metrics
print('Mean Absolute Error',metrics.mean_absolute_error(y_test,y_pred))
```

```
Mean Absolute Error 4.4197278088027652
```

Predicting the Score

In [22]:
```
hours=float(input("Enter the number of Hours"))
percentage= regressor.predict([[hours]])
print("Predicted Percentage:",percentage)
```

```
Enter the number of Hours9.25
Predicted Percentage: [92.91505723]
```

In [ ]: