GRIP : The Sparks Foundation

Data Science and Business Analytics Intern

Batch : March 2022

Author : Pragati Dilip Gawale

Task 1 : Prediction Using Supervised ML

In this task we have to predict the percentage score of a student based on the number og hours studied.Here we can use simple linear regression as there are only two variables involved viz. No.of study hours(Independent) and Score(Dependent)

In [3]:

```
#Importing required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

print("Libraries imported successfully")
```

```
Libraries imported successfully
```

Reading the data

In [6]:

```
url= "http://bit.ly/w-data"
df= pd.read_csv(url)
print("Data imported successfully")
```

```
Data imported successfully
```

Data Exploration

In [7]:

```
df.head()
```

Out[7]:

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

In [8]:

```
df.tail()
```

Out[8]:

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

In [9]:

```
#checking for any missing values

df.isnull().sum()
```

Out[9]:

```
Hours     0
Scores    0
dtype: int64
```

In [10]:

```
df.describe()
```

Out[10]:

	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 [11]:

```
df.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
```

In [13]:

```
#Checking for the correlation between two variables

df.corr()
```

Out[13]:

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

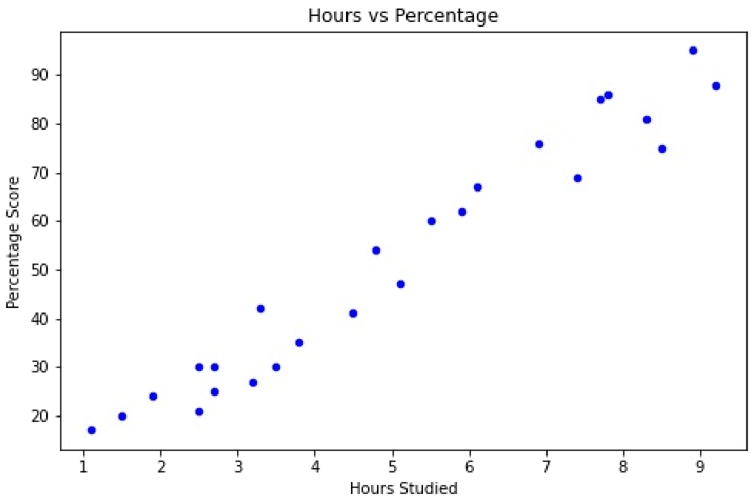
this shows high positive correlation between hours and study

Data Visualization

In [14]:

```
# Scatter plot

df.plot(kind='scatter',x='Hours',y='Scores',color='blue',figsize=(8,5))
plt.title('Hours vs Percentage')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.show()
```



From Scatter plot ,we can say that as no.of hours studied increases ,percentage scores also increases. Positive linear relationship is present.

Modeling the data

In [15]:

```
x=np.asanyarray(df[['Hours']])
y=np.asanyarray(df[['Scores']])

#using train test split to split the data in train and test data
train_x,test_x,train_y,test_y=train_test_split(x,y,test_size=0.2,random_state=2)

regressor=LinearRegression()
regressor.fit(train_x,train_y)
```

```
print("Training of the data completed\n")
print('Coefficients : ',regressor.coef_)
print('Intercept : ',regressor.intercept_)
```

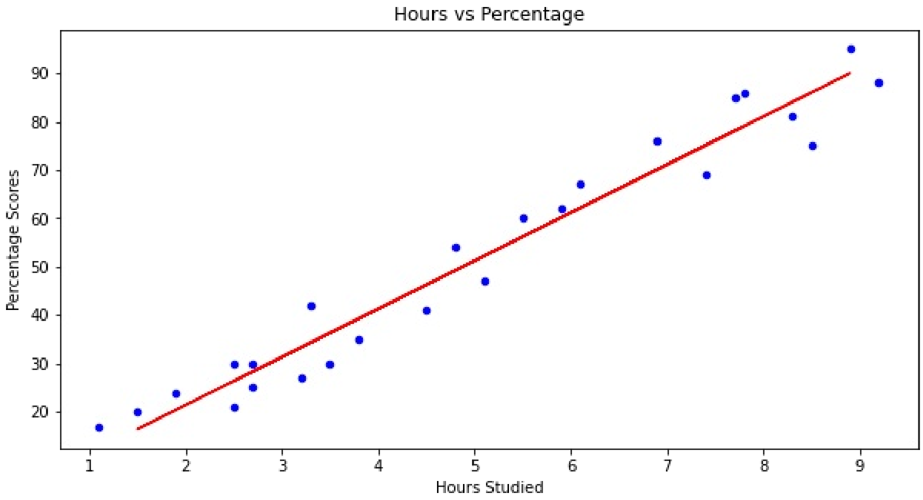
```
Training of the data completed

Coefficients :  [[9.94061514]]
Intercept :  [1.50791048]
```

In [16]:

```
# Plotting the regression line

df.plot(kind='scatter',x='Hours',y='Scores',figsize=(10,5),color='blue')
plt.plot(train_x,regressor.coef_[0]*train_x+regressor.intercept_,color='r')
plt.title('Hours vs Percentage')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Scores')
plt.show()
```



The blue line is the best fit line for the data

Evaluation of the model

In [17]:

```
#using metrics to find mean absolute  error and r2 to see the accuracy

from sklearn import metrics
from sklearn.metrics import r2_score

y_pred=regressor.predict(test_x)
print('Mean Absolute Error : {}'.format(metrics.mean_absolute_error(y_pred,test_y)))
print('R2-score : %.2f' % r2_score(y_pred,test_y))
```

```
Mean Absolute Error : 4.877039354964476
R2-score : 0.98
```

Mean Absolute Error : It is mean of absolute value of errors r2-score : It is not the error but its the metric for accuracy for the model. Higher the r2 value higher is the accuracy of model.Best score is 1 Here r2-score is 0.98 which is quiet good

Predicting the Score

In [23]:

```
hours=9.25
predicted_score=regressor.predict([[hours]])

print(f'No.of hours = {hours}')
print(f'predicted Score = {predicted_score[0]}')
```

```
No.of hours = 9.25
predicted Score = [93.45860056]
```

If a student studies for 9.25 hours per day he/she will score 93.45860056

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js