Presented by Priyanka Fonia, a Data Science and Business Analytics Intern at the Sparks Foundation </n>

From #GRIPMAR2021

TASK 1: Prediction Using Supervised ML

Objective:

Predict the percentage of a student based on the no. of study hours. What will be the predicted score if a student studies for 9.25 hrs/ day?

About the task: This task is Supervised Machine Learning(ML) and such kind of learning includes the training of the model on a labelled dataset that includes both training and validation of datasets. The Labelled dataset is one that has both input and output parameters.

Step 1: Importing the relavant libraries

In [50]:

```
#Importing the relevant libraries

#numpy is numerical python library and includes a multi-dimentional array and matrix
import numpy as np

#Pandas is a library of python used for data manipulation and analysis.
import pandas as pd

#matplotlib. pyplot is a collection of functions that make matplotlib work like MATL
import matplotlib.pyplot as plt

#it is a data visualization library
import seaborn as sns

#%matplotlib allows to add plots to the browser interface
%matplotlib inline

#Sklearn is an efficient tools for machine learning and statistical modeling includi
#classification, regression, clustering and dimensionality reduction
from sklearn.linear_model import LinearRegression #
sns.set()
```

Step 2: Loading, Reading or Understanding the data

Loading the Data for further evaluation

In [51]:

```
#Loading of data
data_url = 'http://bit.ly/w-data'
data_scoreprediction  = pd.read_csv(data_url)
data_scoreprediction.head()
```

Out[51]:

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

STEP 3: Understanding the given data

Shape of the given data(Number of Rows and Columns)

In [52]:

```
data_scoreprediction.shape #We are having 25 rows and 2 columns
```

Out[52]:

```
(25, 2)
```

In [53]:

```
data_scoreprediction.columns
```

Out[53]:

```
Index(['Hours', 'Scores'], dtype='object')
```

Describe method is used for calculating the DataFrame.

In [54]:

```
data_scoreprediction.describe
```

Out[54]:

```
<bound method NDFrame.describe of     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>
```

STEP 4: Understanding the data types

Hours is having float data type and Scores is having Integer data type

In [55]:

```
data_scoreprediction.dtypes
```

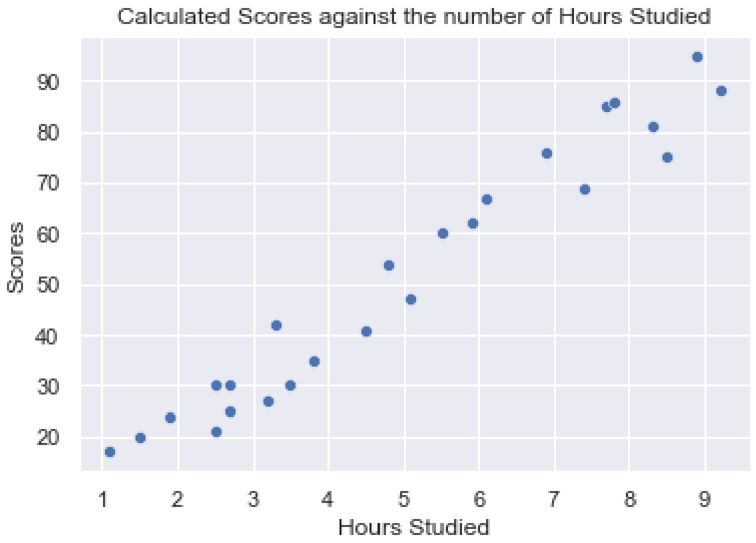
Out[55]:

```
Hours     float64
Scores      int64
dtype: object
```

STEP 5: Visualization of the given data

In [56]:

```
#visualizing data in scatterplot
sns.scatterplot(x='Hours', y='Scores', data=data_scoreprediction)
plt.title('Calculated Scores against the number of Hours Studied')
plt.xlabel('Hours Studied')
plt.ylabel('Scores')
plt.show()
```



From the above graph, it can be depicted that there is a positive and linear relation between Number of hours studied and the score precentage.

STEP 6: Preparing the data

In [57]:

```
x=data_scoreprediction.iloc[:, :-1].values
y=data_scoreprediction.iloc[:,1].values
```

STEP 7: Splitting of data into training and testing tests </n>

Training the data usually includes splitting the data into 80:20 i.e., 80% as training data and rest as testing data. In training data, we feed input as well as output for 80% data </n>

At the time of testing, the input is fed from the remaining 20% of data, the model will predict some value and we will compare it with actual output and calculate the accuracy.

In [58]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state =
#here test size 0.2 means, 20% of data is included in test and rest 80% in training
```

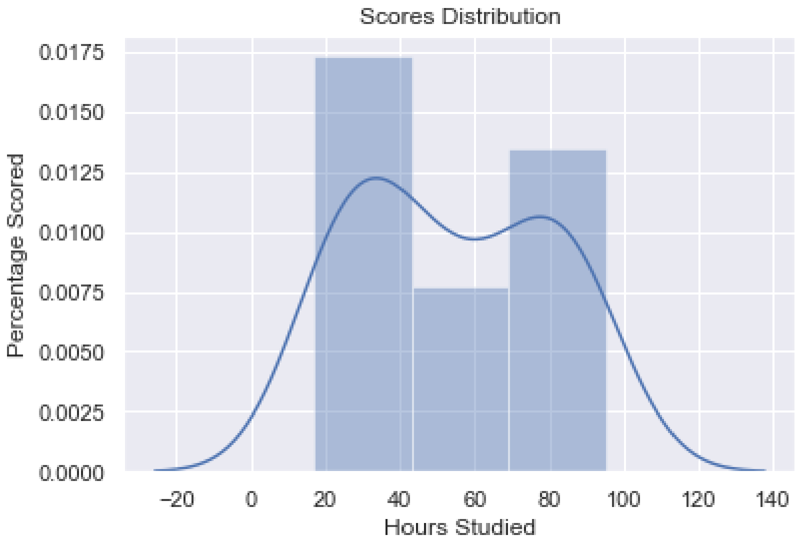
In [59]:

```
sns.distplot(y_train, kde=True)
plt.title('Scores Distribution')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Scored')
```

```
C:\Users\priya\Documents\Anaconda\Anaconda\lib\site-packages\seaborn\distributions.p
y:2551: FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level funct
ion with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[59]:

```
Text(0, 0.5, 'Percentage Scored')
```



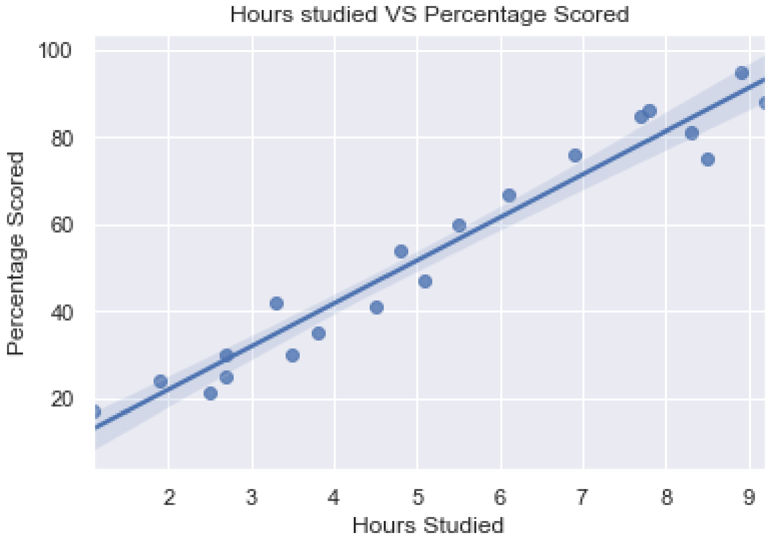
In [60]:

```
sns.regplot(x_train, y_train)
plt.title('Hours studied VS Percentage Scored')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Scored')
```

```
C:\Users\priya\Documents\Anaconda\Anaconda\lib\site-packages\seaborn\_decorators.py:
36: FutureWarning: Pass the following variables as keyword args: x, y. From version
0.12, the only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[60]:

```
Text(0, 0.5, 'Percentage Scored')
```



In [61]:

```
#Linear regression is a way understand the relationship between two variables. These
#Where Y is the dependent variable (plot on the Y axis), X is the independent variab

training=LinearRegression()
training.fit(x_train, y_train)
y_pred=training.predict(x_test)
df=pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
df.head(5)
```

Out[61]:

	Actual	Predicted
0	20	16.884145
1	27	33.732261
2	69	75.357018

	Actual	Predicted
3	30	26.794801
4	62	60.491033

Step 8: Model Evaluation

In [62]:

```
#mean absoulute error
from sklearn.metrics import mean_absolute_error
print("Mean absolute error: ", mean_absolute_error(y_test,y_pred))
```

```
Mean absolute error:  4.183859899002975
```

In [63]:

```
#r2square
from sklearn.metrics import r2_score
print("Prediction error:", r2_score(y_test, y_pred) )
```

```
Prediction error: 0.9454906892105356
```

STEP 9: Prediction of Future Data </n>

Predicting the score of a student when he studies for 9.25 hours per day.

In [64]:

```
hours=[[9.25]]
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(x_train,y_train)
pred=reg.predict(hours)

#Reg is a Python library that provides generic function support to Python.
#It help you build powerful registration and configuration APIs for your application

print("Score obtained by the student if he studies for 9.25 hours per day={}".format
```

```
Score obtained by the student if he studies for 9.25 hours per day=93.69173248737538
```

In [ ]: