The Sparks Foundation Data Science and Business Analytics (Grip 2021) Task 1 : Prediction using Supervised ML Problem : predict the percentage of students based on no. of study hours. Author: Venepally Tharun

1. Importing all Libraries required

In [1]:
```
#importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns #for data visualisation.
```

1. Reading Dataset

In [2]:
```
#Reading data from link
url = r"http://bit.ly/w-data"
df = pd.read_csv(url)
print("Data imported successfully")
```

```
Data imported successfully
```

In [3]:
```
#printing first 10 rows of the data set.
df.head(10)
```

Out[3]:

	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 [4]:
```
#checking the number of rows and columns.
df.shape
```

Out[4]:
```
(25, 2)
```

In [5]:
```
df.describe()
```

Out[5]:

	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 [6]:
```
df.info()
```

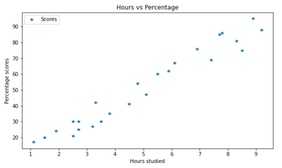
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   25 non-null    float64
 1   Scores  25 non-null    int64  
dtypes: float64(1), int64(1)
memory usage: 528.0 bytes
```

In [7]:
```
#check if there is any null value in the dataset.
df.isnull().sum()
```

Out[7]:
```
Hours     0
Scores    0
dtype: int64
```

3 . DATA VISUALISATION

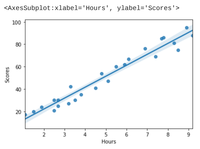
In [8]:
```
#Plotting the graph to see the relation and distribution of datapoints.
df.plot(x = 'Hours', y = 'Scores', figsize = (9,5), style = '*')
plt.title('Hours vs Percentage')
plt.xlabel('Hours studied')
plt.ylabel('Percentage scores')
plt.show()
```



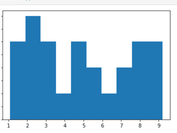
from the above graph we can see that there is positive linear relation between the no of hours studied and percentage scores.

In [9]:
```
sns.regplot( x = 'Hours', y = 'Scores', data = df)
```

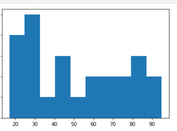
Out[9]:
```
<AxesSubplot:xlabel='Hours', ylabel='Scores'>
```



In [10]:
```
#Distribution of hours data
plt.hist(x = 'Hours', data = df)
plt.show()
```



In [11]:
```
#Distribution of scores data
plt.hist(x = 'Scores', data = df)
plt.show()
```



4 . DATA PREPARATION

In [12]:
```
#dividing the datasets int attributes(inputs) and labels(columns)
x = df.iloc[:,:-1].values
y = df.iloc[:,1].values
```

slpit this data into train and testdata using the train_test_split() method from the skicit learn library.

In [13]:
```
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 = 0)
```

5 . TRAINING THE MODEL we have to train our data using Linear regression algorithm and check the values for test data

In [14]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

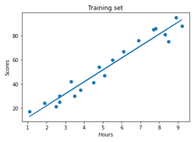
Out[14]:
```
LinearRegression()
```

plot the regression line in the scatter

In [15]:
```
r_line = lr.coef_*x+lr.intercept_
```

In [16]:
```
#visualising the training dataset
plt.scatter(x_train, y_train)
plt.title('Training set')
plt.plot(x_train,lr.predict(x_train))
plt.xlabel("Hours")
plt.ylabel("Scores")
```

Out[16]:
```
Text(0, 0.5, 'Scores')
```



6 . MAKING PREDICTIONS

we have done tarining and we have to make some predictions.

In [17]:
```
y_pred = lr.predict(x_test)
y_pred
```

Out[17]:
```
array([16.88414476, 33.73226078, 75.357018  , 26.79480124, 60.49103328])
```

compare actual values with predicted values

In [18]:
```
df = pd.DataFrame({"Actual":y_test,"Predicted":y_test})
df
```

Out[18]:

	Actual	Predicted
0	20	20
1	27	27
2	69	69
3	30	30
4	62	62

In [19]:
```
#score for test data
accuracy = lr.score(x_test, y_test)
print("Accuracy:",accuracy * 100)
```

```
Accuracy: 94.54906892105356
```

what is the predicted score if student studies for 9.25 hours per day?

In [20]:
```
# we can test for any input
# here we are calculating the score for 9.25 hours
hours = [[9.25]]
pred = lr.predict(hours)
pred
```

Out[20]:
```
array([93.69173249])
```

1. MODEL EVALUATION

In this step we have to evaluate the performance of algorithm. we can evaluate this by calculating mean squared error or mean absolute error.

MEAN ABSOLUTE ERROR

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

```
Mean Absolute Error:  4.183859899002975
```

MEAN SQUARED ERROR

In [22]:
```
from sklearn import metrics
MSE = metrics.mean_squared_error(y_test,y_pred)
print("Mean Squared Error: ", MSE)
```

```
Mean Squared Error:  21.5987693072174
```

Small value of Mean Absolute Error states that the model is quite good.

THANK YOU.

In [ ]:
```

```