GRADUATE ROTATIONAL INTERNSHIP PROGRAM (GRIP)

THE SPARK FOUNDATION

NAME: OMKAR SALUNKHE.

TASK1:PREDICTION USING SUPERVISED ML

Predict the percentage of student on basis of how many hour in a day they study.

Importing Libraries

In [24]:
```
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
%matplotlib inline
```

Import Dataset

In [2]:
```
data_df=pd.read_csv("my_data")
data_df
```

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_df.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

To check shape of daaset ,hoe many no of columns and rows containg dataset

In [4]:
```
data_df.shape
```

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

In [5]:
```
data_df.columns
```

Out[5]:
```
Index(['Hours', 'Scores'], dtype='object')
```

In [6]:
```
print('The give data contains columns :',len(data_df.columns))
```

```
The give data contains columns : 2
```

check the information about our dataset

In [7]:
```
data_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 [8]:
```
data_df.describe()
```

Out[8]:

	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

checking for missing or null value are present or not

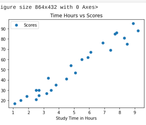
In [9]:
```
data_df.isna().sum()
```

Out[9]:
```
Hours     0
Scores    0
dtype: int64
```

ploting a graph to get clear idea about our dataset

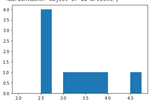
In [10]:
```
plt.figure(figsize=(12,6))
data_df.plot(x="Hours", y="Scores" , style="o")
plt.title("Time Hours vs Scores")
plt.xlabel("Study Time in Hours")
plt.ylabel("Scores")
plt.legend()
plt.show()
```

```
<Figure size 864x432 with 0 Axes>
```



In [11]:
```
plt.hist(data_df.Hours , bins=np.arange(2,5,0.25))
```

Out[11]:
```
(array([0., 0., 4., 0., 1., 1., 1., 1., 1., 0., 0., 1.]),
 array([2.  , 2.25, 2.5 , 2.75, 3.  , 3.25, 3.5 , 3.75, 4.  , 4.25, 4.5 ,
        4.75]),
 <BarContainer object of 11 artists>)
```



correlation is useful to find out relation among them.

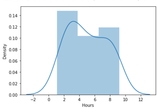
In [12]:
```
data_df.corr()
```

Out[12]:

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

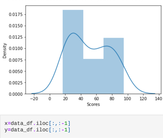
In [13]:
```
sns.distplot(data_df["Hours"])
```

Out[13]:
```
<AxesSubplot:xlabel='Hours', ylabel='Density'>
```



In [14]:
```
sns.distplot(data_df["Scores"])
```

Out[14]:
```
<AxesSubplot:xlabel='Scores', ylabel='Density'>
```



In [15]:
```
x=data_df.iloc[:,:-1]
y=data_df.iloc[:,1:]
```

splitting the dataset into train and test set

In [20]:
```
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)
```

In [26]:
```
y_test
```

Out[26]:

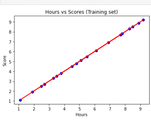
	Hours
5	1.5
2	3.2
19	7.4
16	2.5
11	5.9

In [32]:
```
from sklearn.linear_model import LinearRegression
model= LinearRegression()
model.fit(x_train,y_train)
```

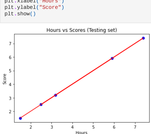
Out[32]:
```
LinearRegression()
```

In [33]:
```
y_pred=model.predict(x_test)
```

In [34]:
```
#visualize the training test result
plt.scatter(x_train,y_train,color="blue")
plt.plot(x_train,model.predict(x_train),color="red")
plt.title("Hours vs Scores (Training set)")
plt.xlabel("Hours")
plt.ylabel("Score")
plt.show()
```



In [35]:
```
plt.scatter(x_test,y_test,color="blue")
plt.plot(x_test,y_pred,color="red")
plt.title("Hours vs Scores (Testing set)")
plt.xlabel("Hours")
plt.ylabel("Score")
plt.show()
```



In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```

```