GRIP: The Sparks Foundation

Data Science and Business Analytics Intern

Author: Wrushabh Narendra Gonnade

Task 1: Prediction using Supervised ML

In [30]:
```
#Importing Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.metrics import r2_score
```

In [2]:
```
stu_data=pd.read_csv('http://bit.ly/w-data')
```

In [3]:
```
stu_data
```

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
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 [4]:

```
#Getting the rows and column
stu_data.shape
```

Out[4]:

```
(25, 2)
```

In [5]:

```
#Gives the Statistical Information
stu_data.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]:

```
#Summary of Dataframe
stu_data.info()
```

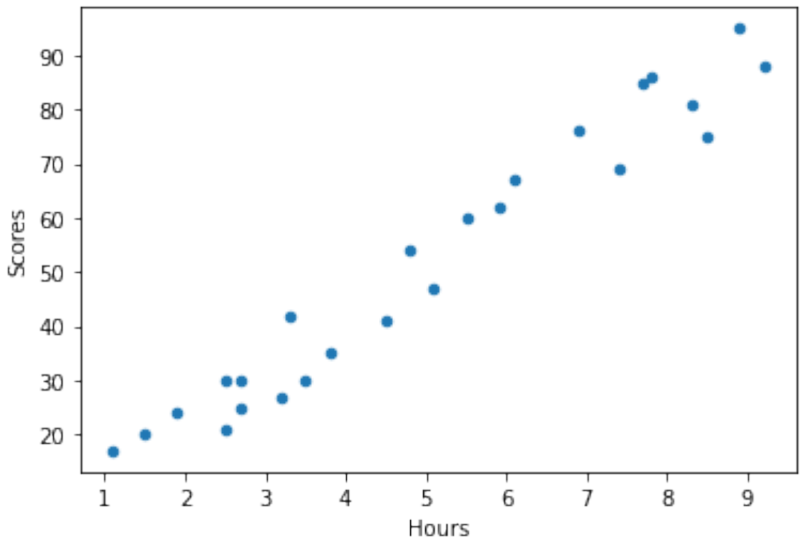
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64
dtypes: float64(1), int64(1)
memory usage: 528.0 bytes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

In [7]:

```
stu_data.plot(kind='scatter',x='Hours',y='Scores')
```

Out[7]:

```
<AxesSubplot:xlabel='Hours', ylabel='Scores'>
```



In [8]:

```
#Corelation Coefficient
stu_data.corr(method='pearson')
```

Out[8]:

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

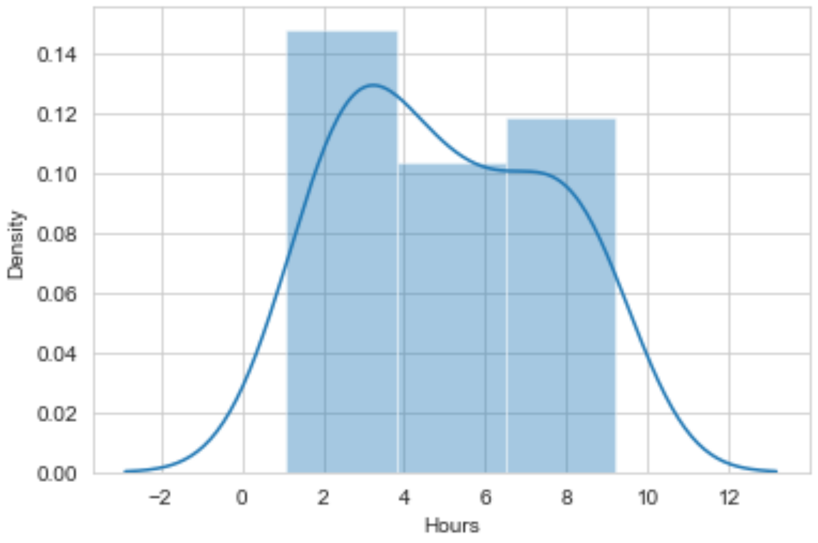
In [35]:

```
#Distribution Model
sns.distplot(stu_data['Hours'])
```

```
/Users/wrushabhgonnade/opt/anaconda3/lib/python3.8/site-packages/seaborn/di
stributions.py:2557: 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 function with similar flexibility) or `histplot` (
an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[35]:

```
<AxesSubplot:xlabel='Hours', ylabel='Density'>
```

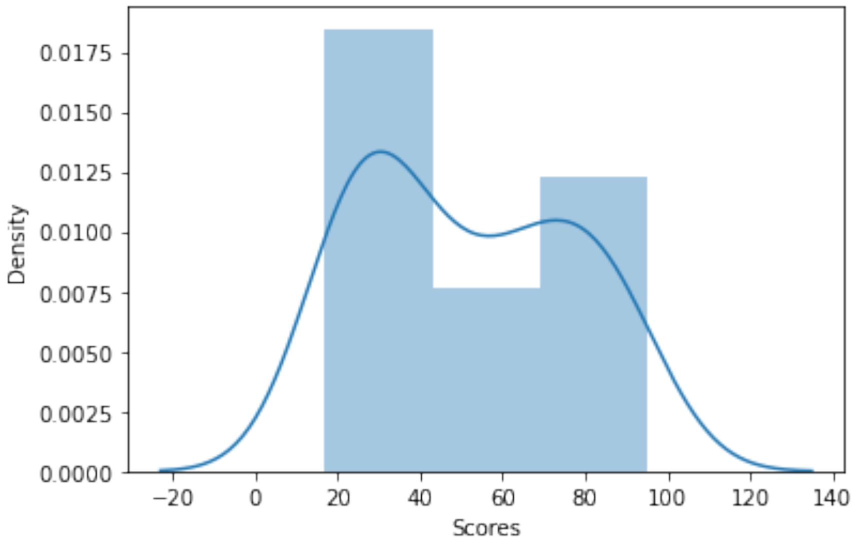


In [10]:

```
sns.distplot(stu_data['Scores'])
```

```
/Users/wrushabhgonnade/opt/anaconda3/lib/python3.8/site-packages/seaborn/di
stributions.py:2557: 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 function with similar flexibility) or `histplot` (
an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[10]:
```
<AxesSubplot:xlabel='Scores', ylabel='Density'>
```



Preparing the data

In [14]:
```
X=stu_data.iloc[:,:-1].values
Y=stu_data.iloc[:,1].values
```

In [15]:
```
from sklearn.model_selection import train_test_split
```

In [16]:
```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_st
```

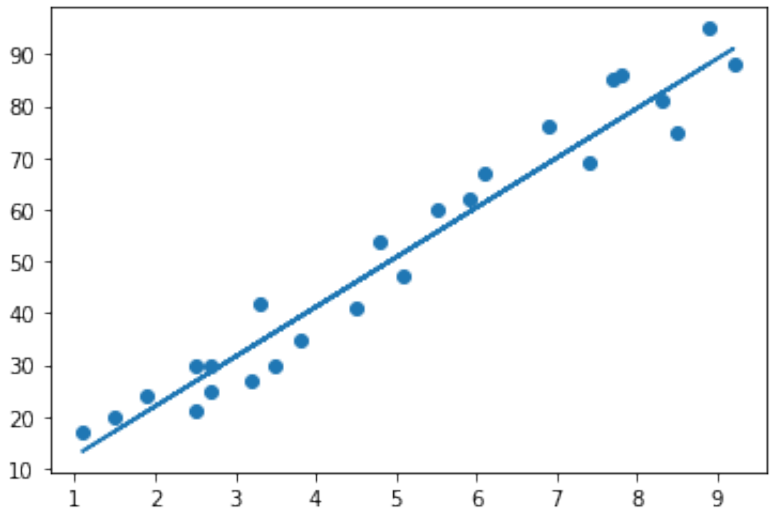
In [17]:
```
from sklearn.linear_model import LinearRegression
```

In [18]:
```
reg=LinearRegression()
reg.fit(X_train,Y_train)
```

Out[18]:
```
LinearRegression()
```

Linear Regression

In [24]:
```
m=reg.coef_
c=reg.intercept_
line=m*X+c
plt.scatter(X,Y)
plt.plot(X,line)
plt.show()
```



In [26]:
```
#Compare Actual vs Predicted Data
y_pred=reg.predict(X_test)
```

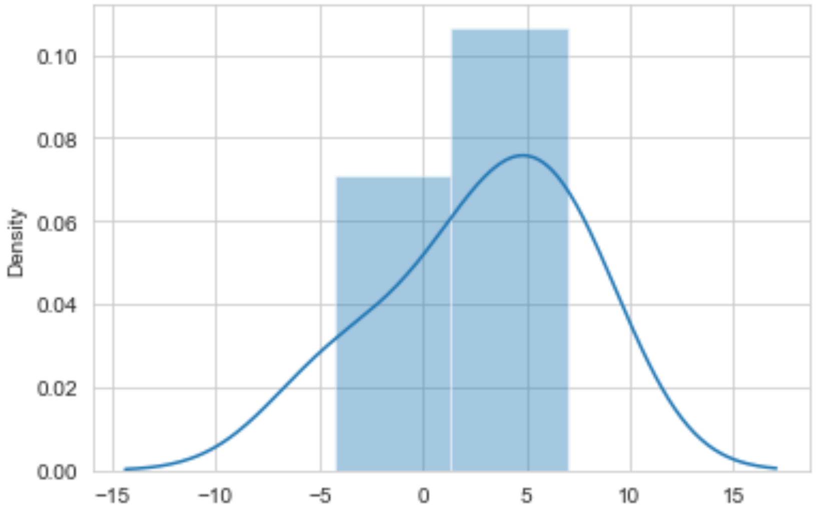
In [27]:
```
act_pred=pd.DataFrame({'Target':Y_test,'Predicted':y_pred})
act_pred
```

Out[27]:

	Target	Predicted
0	95	88.211394
1	30	28.718453
2	76	69.020122
3	35	39.273652
4	17	13.365436

In [28]:
```
sns.set_style('whitegrid')
sns.distplot(np.array(Y_test-y_pred))
plt.show()
```

```
/Users/wrushabhgonnade/opt/anaconda3/lib/python3.8/site-packages/seaborn/di
stributions.py:2557: 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 function with similar flexibility) or `histplot` (
an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Predicted Score if a student studies for 9.25 Hours per day

In [29]:

```
h=9.25
s=reg.predict([[h]])
print('If a Student Studies for {} hours per day he/she will score {} % in
```

```
If a Student Studies for 9.25 hours per day he/she will score [91.56986604]
% in exam.
```

Model Evaluation

In [33]:

```
print('Mean Absolute Error: ',metrics.mean_absolute_error(Y_test,y_pred))
print('R2 Score: ',r2_score(Y_test,y_pred))
```

```
Mean Absolute Error:  4.5916495300630285
R2 Score:  0.971014141329942
```

In [ ]:

In [ ]: