**GRIP @ Sparks Foundation**

**Data Science and Business Analytics Internship**

**Author:Rittik Hazra**

**Task 1: Prediction Using Supervised ML**

In [ ]:

*#Import required Libraries*

**import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

In [125]:

*#Importing dataset*

**data1=pd.read\_csv("C:\\Users\\ritti\\Desktop\\sparks project1.csv")**

**data1.head()**

Out[125]:

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

Explor the data

In [9]:

**print(data1.shape)**

(25, 2)

In [140]:

**data1.describe()**

Out[140]:

|  | **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]:

**data1.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

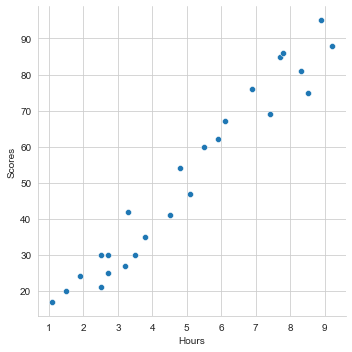
1 Scores 25 non-null int64

dtypes: float64(1), int64(1)

memory usage: 528.0 bytes

In [128]:

**sns.relplot(kind="scatter",x="Hours",y="Scores",data=data1);**



After plotting the graph, we have observed to that scores is incresing as well as study hours,which is good significant of correct data.

In [31]:

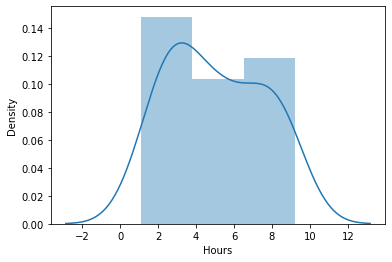
**data1.corr(method="spearman")**

Out[31]:

|  | **Hours** | **Scores** |
| --- | --- | --- |
| **Hours** | 1.000000 | 0.971891 |
| **Scores** | 0.971891 | 1.000000 |

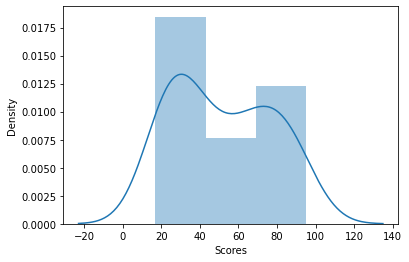
In [42]:

**sns.distplot(data1['Hours']);**



In [43]:

**sns.distplot(data1['Scores']);**



Now we have to prepared data for the model

In [109]:

**x=data1.iloc[:, :-1].values**

**print(x)**

**y=data1.iloc[:, 1].values**

**print(y)**

[[2.5]

[5.1]

[3.2]

[8.5]

[3.5]

[1.5]

[9.2]

[5.5]

[8.3]

[2.7]

[7.7]

[5.9]

[4.5]

[3.3]

[1.1]

[8.9]

[2.5]

[1.9]

[6.1]

[7.4]

[2.7]

[4.8]

[3.8]

[6.9]

[7.8]]

[21 47 27 75 30 20 88 60 81 25 85 62 41 42 17 95 30 24 67 69 30 54 35 76

86]

we have devide the data into trainning and testing the model

In [111]:

**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=50)**

Trainning the algorithm

In [132]:

**from sklearn.linear\_model import LinearRegression**

**reg=LinearRegression()**

**reg.fit(x\_train,y\_train)**

**print("trainning complete")**

trainning complete

now the model is ready to test

In [123]:

**m=reg.coef\_**

**m**

**c=reg.intercept\_**

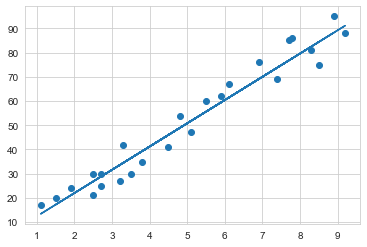
**c**

**line=m\*x+c**

**plt.scatter(x,y)**

**plt.plot(x,line)**

**plt.show()**



Checking the accuracy of the model

In [124]:

**y\_pred=reg.predict(x\_test)**

**actual\_predicted=pd.DataFrame({'Target':y\_test,'Predicted':y\_pred})**

**actual\_predicted**

Out[124]:

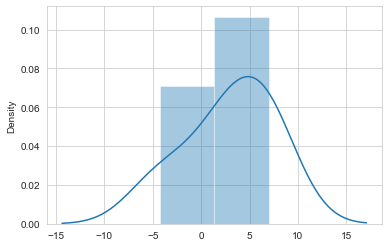
|  | **Target** | **Predicted** |
| --- | --- | --- |
| **0** | 95 | 88.211394 |
| **1** | 30 | 28.718453 |
| **2** | 76 | 69.020122 |
| **3** | 35 | 39.273652 |
| **4** | 17 | 13.365436 |

In [116]:

**sns.set\_style('whitegrid')**

**sns.distplot(np.array(y\_test-y\_pred))**

**plt.show()**



In [117]:

*#the predicted score if a student studies for 9.25hrs/day*

**h=9.25**

**b=reg.predict([[h]])**

**print("If a student studies for {} hours per day he/she will score {} % in exam.".format(h,b))**

If a student studies for 9.25 hours per day he/she will score [91.56986604] % in exam.

Evaluating the model

In [138]:

**from sklearn import metrics**

**print("Mean Absolute Error:",**

**metrics.mean\_absolute\_error(y\_test,y\_pred))**

Mean Absolute Error: 4.5916495300630285

Thank You