GRIP: The Sparks Foundation

Data Science and Business Analytics Intern

Author:- Sudeshna Saha

In [1]:

```
#import all necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Import the dataset

In [2]:

```
df=pd.read_csv("http://bit.ly/w-data")
```

In [3]:

```
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

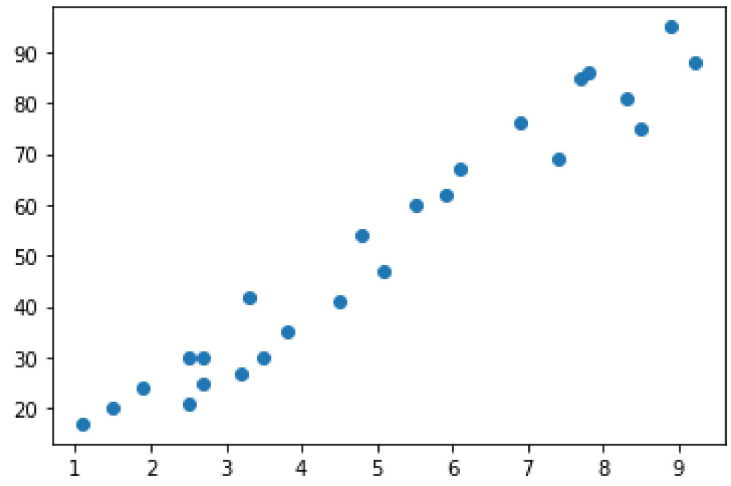
Plotting our dataset

In [4]:

```
plt.scatter(df["Hours"],df["Scores"])
```

Out[4]:

```
<matplotlib.collections.PathCollection at 0x250d70e6ca0>
```

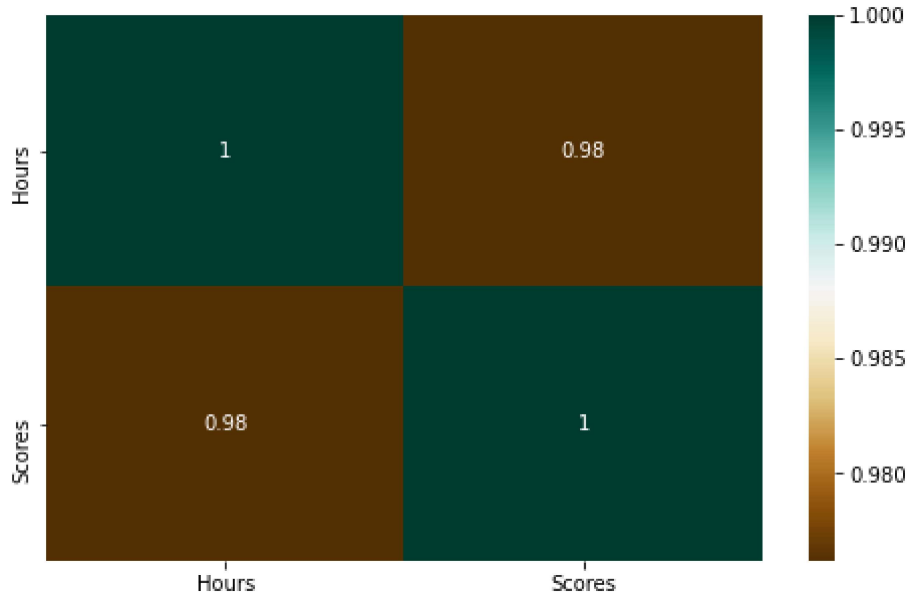


In [5]:

```
#check the correlation of the dataset
plt.figure(figsize=(8,5))
sns.heatmap(df.corr(),cmap="BrBG",annot=True)
```

Out[5]:

```
<AxesSubplot:>
```



In [6]:

```
#shape of the dataset
df.shape
```

Out[6]:

```
(25, 2)
```

In [17]:

```
df.describe()
```

Out[17]:

	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 [18]:

```
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
```

Linear Regression Model

In [7]:

```
#preparing the data
x=df.drop(["Scores"],axis='columns',inplace=False)
y=df["Scores"]
```

In [8]:

```
#for machine learning model import some libraries
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

In [9]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [10]:

```
lgr=LinearRegression()
model=lgr.fit(x_train,y_train)
pred=lgr.predict(x_test)
```

In [11]:

```
pred
```

Out[11]:

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

In [12]:

```
#comparing actual vs predicted
df2=pd.DataFrame({'Actual':y_test,'Predicted':pred})
df2
```

Out[12]:

	Actual	Predicted
5	20	16.884145
2	27	33.732261
19	69	75.357018
16	30	26.794801
11	62	60.491033

Model Evaluation

In [13]:

```
#mean squared error value
mse=mean_squared_error(y_test,pred)
mse
```

Out[13]:

```
21.5987693072174
```

In [14]:

```
#R squared value
r2=model.score(x_train,y_train)
r2
```

Out[14]:

```
0.9515510725211552
```

In [15]:

```
#we also can check with unknown data
hours=[9.25]
own_pred=(lgr.predict(pd.DataFrame(hours)))
```

In [16]:

```
own_pred
```

Out[16]:

```
array([93.69173249])
```

In [ ]:

```

```