Python for data analysis

Simple linear Regression

In [149]:

```
import numpy as np  #import libraries
import pandas as pd
import seaborn as sns
import sklearn as sl
import matplotlib.pyplot as plt
%matplotlib inline
```

In [150]:

```
data=pd.read_excel(r'F:\dataset.xlsx')    ## import dataset
```

```
In [151]: data
```

Out[151]:

	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30
5	3.5	34
6	5.3	34
7	4.2	22
8	1.5	53
9	2.5	24
10	4.1	12
11	9.7	47
12	6.7	27
13	8.5	23
14	4.9	30
15	7.8	34
16	9.3	34
17	7.9	22
18	5.5	53
19	6.7	24
20	3.5	54
21	6.7	30
22	7.8	34
23	9.6	76
24	8.1	34

Basic statistic of data

In [152]:

```
print(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: 464.0 bytes
None
```

Given dataset contain 25 entries and there is no any null value present in data.

In [153]:

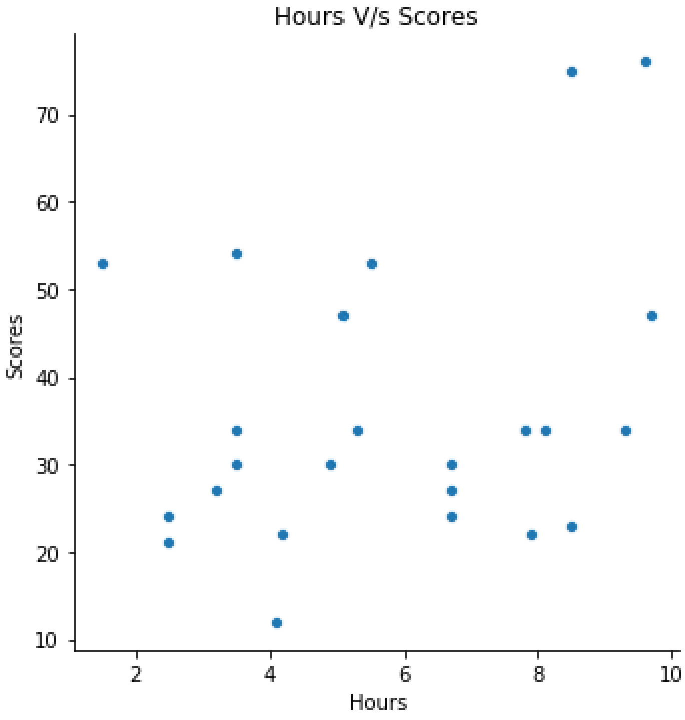
```
print(data.describe())    ##Description of data
```

```
           Hours     Scores
count  25.000000  25.000000
mean    5.864000  36.040000
std     2.470641  16.084361
min     1.500000  12.000000
25%     3.500000  24.000000
50%     5.500000  34.000000
75%     7.900000  47.000000
max     9.700000  76.000000
```

Vizualization

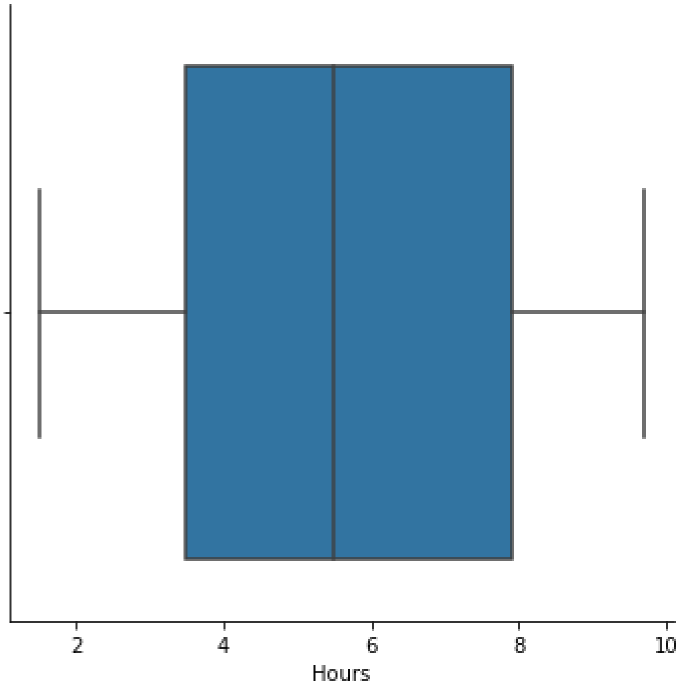
In [154]:

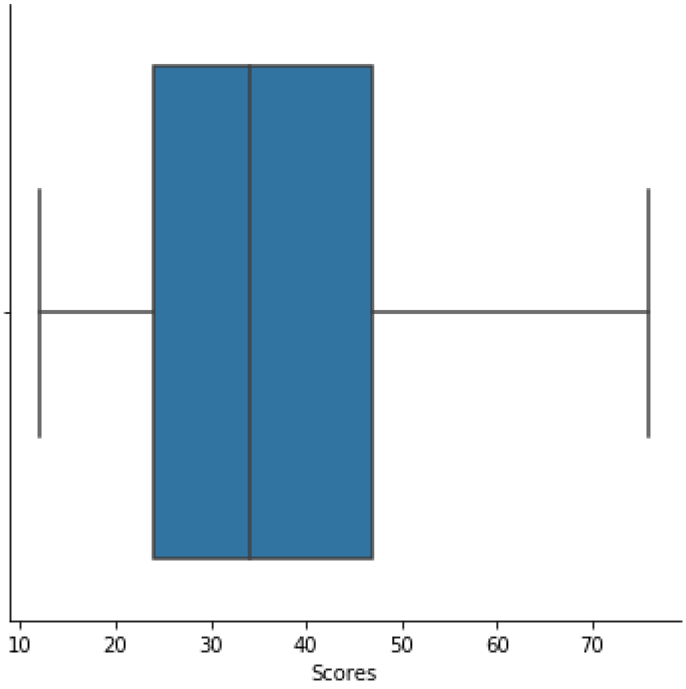
```
sns.relplot(x='Hours', y='Scores',data=data) ## vizualization of data
plt.title("Hours V/s Scores")
plt.show()
```



In [155]:
```
sns.catplot("Hours",data=data,kind='box')
sns.catplot("Scores", data=data,kind='box')
```

Out[155]:
```
<seaborn.axisgrid.FacetGrid at 0x11c84fb0>
```



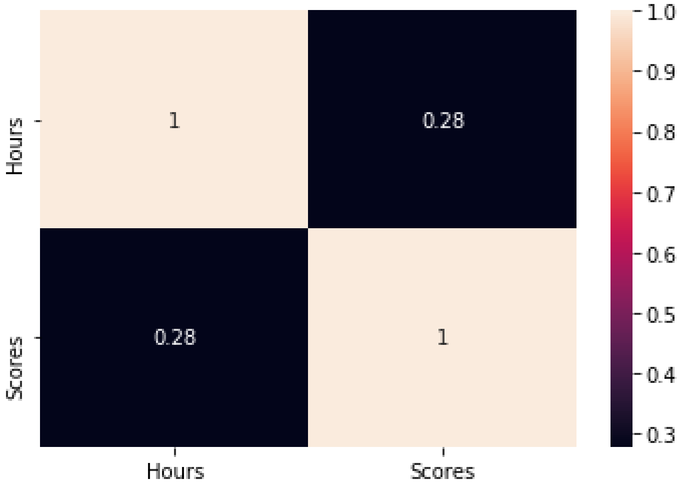


In [156]:
```
cor=data.corr()     ## Correlation of data
cor
```

Out[156]:

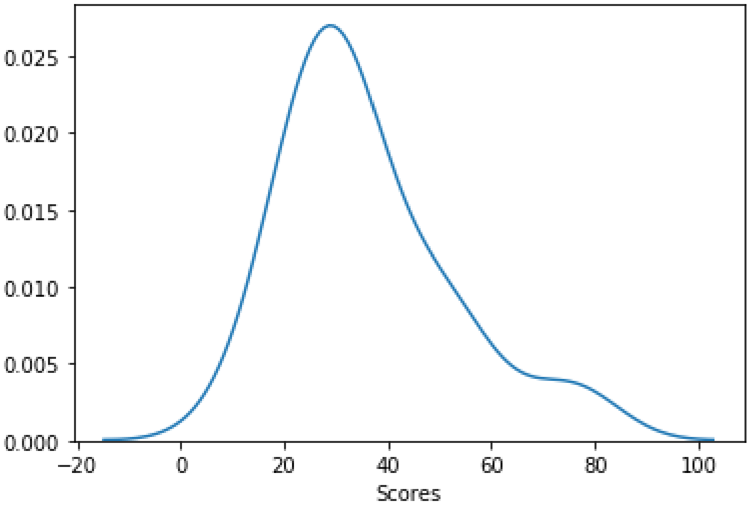
	Hours	Scores
Hours	1.00000	0.27779
Scores	0.27779	1.00000

In [157]:
```
sns.heatmap(cor,annot=True)
plt.show()
```



In [158]:
```
 sns.distplot(data['Scores'],hist=False)    ## plotting the distribution of sc
ores of data
```

Out[158]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x11be85b0>
```



Preparing the data

In [159]:
```
x=data.iloc[:,:-1].values
y=data.iloc[:,1].values
```

Train test split

In [160]:
```
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 [161]:

```
from sklearn.linear_model import LinearRegression

reg=LinearRegression()
model=reg.fit(x_train,y_train)
```

In [162]:

```
print(model.coef_),print(model.intercept_)
```

```
[2.0095554]
25.27543866262086
```

Out[162]:

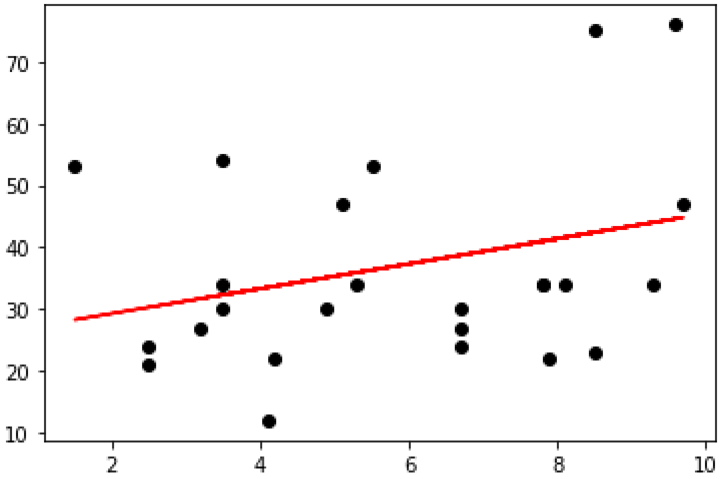
```
(None, None)
```

The best fit of line is

Scores= 12.860+4.577*Hours

In [163]:

```
line=model.intercept_+model.coef_*x  ##Plotting the regression line
plt.scatter(x,y,color='black')
plt.plot(x,line,color='red')
plt.show()
```



Prediction

In [164]:

```
y_pred=model.predict(x_test)
y_pred
```

Out[164]:

```
array([32.30888256, 31.70601594, 38.73945985, 43.96430389, 44.76812605])
```

In [165]:

```
pd.DataFrame({'Actual':y_test,'predict':y_pred})   ##comparing Actual value versus Predict value
```

Out[165]:

	Actual	predict
0	34	32.308883
1	27	31.706016
2	24	38.739460
3	34	43.964304
4	47	44.768126

In [166]:

```
Hours=10.25
Predict_Score=model.predict([[Hours]])
Predict_Score
print('No of Hours=8.5')
print("Predict Score=",format(Predict_Score[0]))
```

```
No of Hours=8.5
Predict Score= 45.87338151868674
```

In [167]:

```
from sklearn import metrics   ##mean square error
print('mean absolute error:',round(metrics.mean_absolute_error(y_test,y_pred)),3)
print('mean Squared error:',round(metrics.mean_squared_error(y_test,y_pred)),3)
print('Root mean squared error:',round(np.sqrt(metrics.mean_squared_error(y_test,y_pred)),3))
```

```
mean absolute error: 7.0 3
mean Squared error: 69.0 3
Root mean squared error: 8.325
```

The value of root mean squared error is 8.325 which is greater than 10% of the mean percentage of all the students(36.04)

In [168]:

```
from scipy import stats  # R- squared value
slope,inter,r,p,std=stats.linregress(y_test,y_pred)
print('R-squared Value:',r)
```

```
R-squared Value: 0.5724771973429205
```

Conclusion: Here R- Squared value is 0.5724 means the model explains 57.24% of variability in the dependent variable (Score) by independent variable (Hours).

```
In [ ]:
```