The Sparks Foundation Grip Internship July 2022

Data Science and Business Analytics Tasks

Intern Information::

Name: Garima Sharma

Domain : Python and Data Science

Qualification : Master of Engineering in Computer Science and Engineering.

Scource: Linkedln

Task 1

Prediction Using Supervised ML (Level: Begineer)

Problem Statement :

1) To predict the percentage of a student based on the study hours.

2) Algorithm : Linear Regression [2 variables]

3) Language : Python

4) Predict : What will be predicted score if a student studies 9.25/hrs a day??

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.model_selection import train_test_split
%matplotlib inline
```

```
df = pd.read_csv("/content/drive/MyDrive/studentscores.csv")
print("File read successful!!")
df
```

```
File read successful!!
```

	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

```
print("Shape of file = ",df.shape)
print("Size of file = ",df.size)
```

```
Shape of file =  (25, 2)
Size of file =  50
```

```
total_columns=pd.DataFrame(df.columns)
total_columns.T
```

	0	1
0	Hours	Scores

```
print("Transpose structure rows to columns and columns to rows transformation")
df.T
```

```
Transpose structure rows to columns and columns to rows transformation
```

	0	1	2	3	4	5	6	7	8	9	...	15	16	17
Hours	2.5	5.1	3.2	8.5	3.5	1.5	9.2	5.5	8.3	2.7	...	8.9	2.5	1.9
Scores	21.0	47.0	27.0	75.0	30.0	20.0	88.0	60.0	81.0	25.0	...	95.0	30.0	24.0

2 rows × 25 columns

```
describe = pd.DataFrame(df.describe())
print("The satistical description :")
describe
```

```
The satistical description :
```

	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

```
df.info()
```

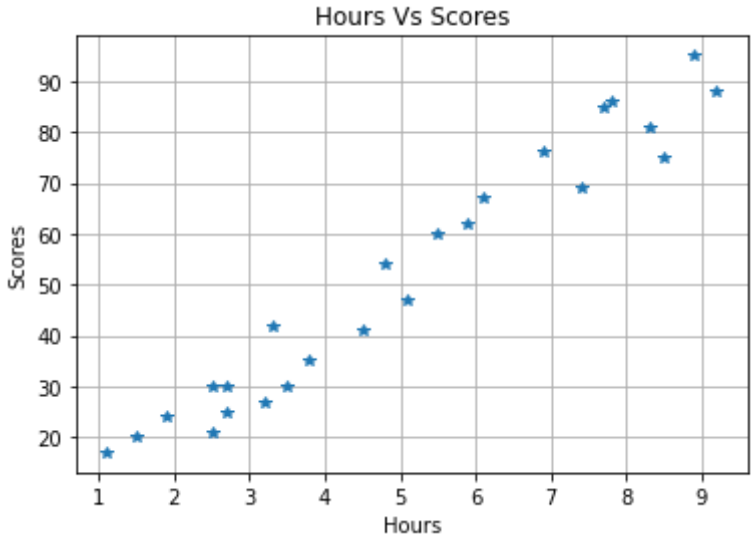
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 528.0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

```
df.isnull().sum()
```

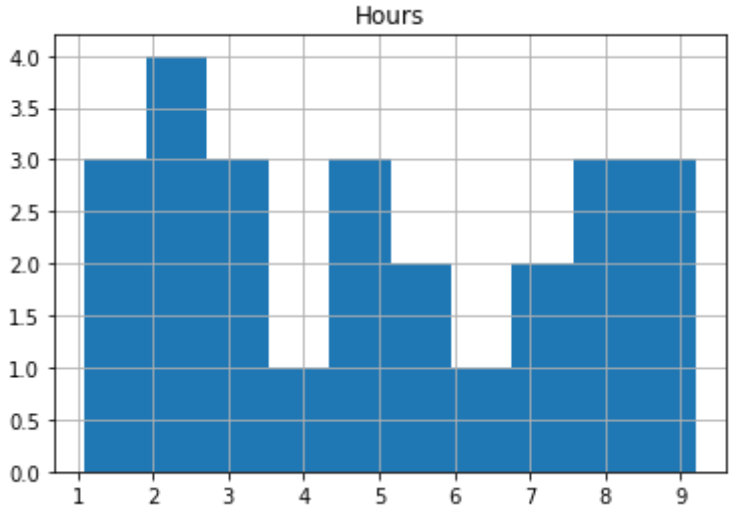
```
Hours     0
Scores    0
dtype: int64
```

Data Visulization

```
x = df["Hours"]
y = df["Scores"]
plt.title("Hours Vs Scores")
plt.xlabel("Hours")
plt.ylabel("Scores")
plt.plot(x,y,"*")
plt.grid()
plt.show()
```

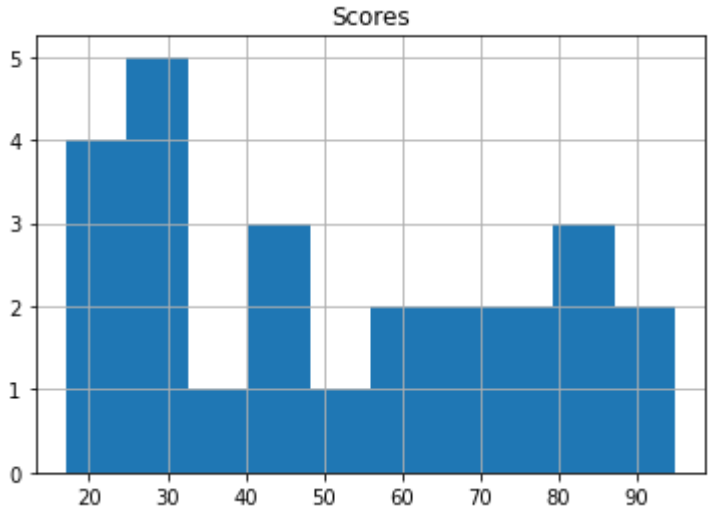



```
#student studyin for hours
x = df["Hours"]
y = df["Scores"]
plt.title("Hours")

plt.hist(x)
plt.grid()
plt.show()
```




```
#number of students with score
```

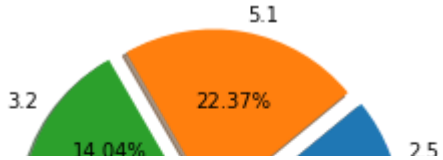
```
x = df["Scores"]

plt.title("Scores")

plt.hist(y)
plt.grid()
plt.show()
```



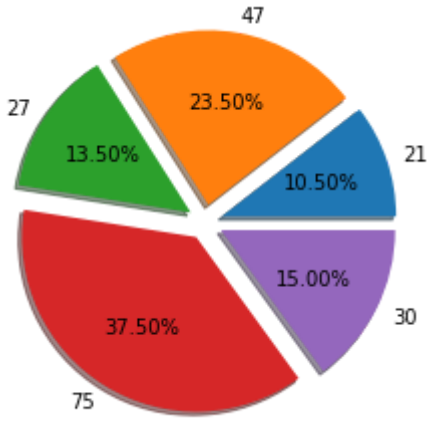

```
df_1 = pd.DataFrame(df[0:5])
df_1
```

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

```
y = df_1["Hours"]
myexplode = [0.1,0.1,0.1,0.1,0.1]
mylabels =[2.5,5.1,3.2,8.5,3.5]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



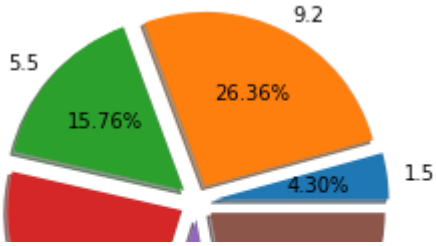

```
y = df_1["Scores"]
myexplode = [0.1,0.1,0.1,0.1,0.1]
mylabels =[21,47,27,75,30]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



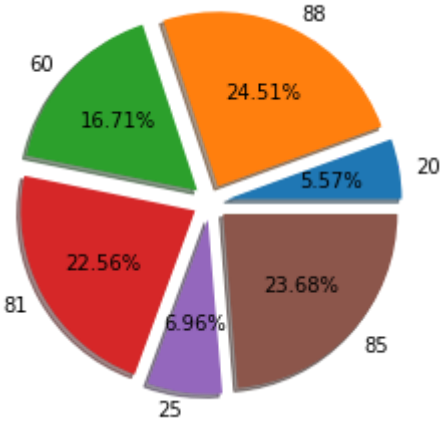

```
df_2 = pd.DataFrame(df[5:11])
df_2
```

	Hours	Scores
5	1.5	20
6	9.2	88
7	5.5	60
8	8.3	81
9	2.7	25
10	7.7	85

```
y = df_2["Hours"]
myexplode = [0.1,0.1,0.1,0.1,0.1,0.1]
mylabels =[1.5,9.2,5.5,8.3,2.7,7.7]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



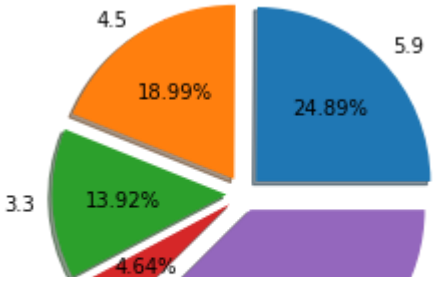

```
y = df_2["Scores"]
myexplode = [0.1,0.1,0.1,0.1,0.1,0.1]
mylabels =[20,88,60,81,25,85]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



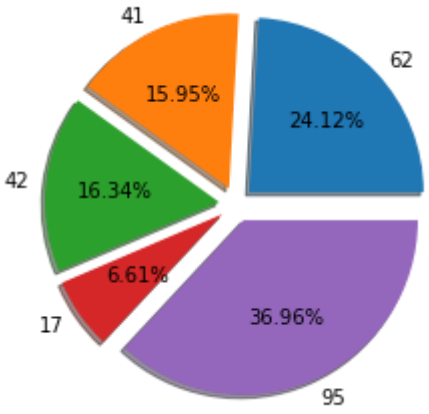

```
df_3 = pd.DataFrame(df[11:16])
df_3
```

	Hours	Scores
11	5.9	62
12	4.5	41
13	3.3	42
14	1.1	17
15	8.9	95

```
y = df_3["Hours"]
myexplode = [0.1,0.1,0.1,0.1,0.1]
mylabels =[5.9, 4.5, 3.3, 1.1, 8.9]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



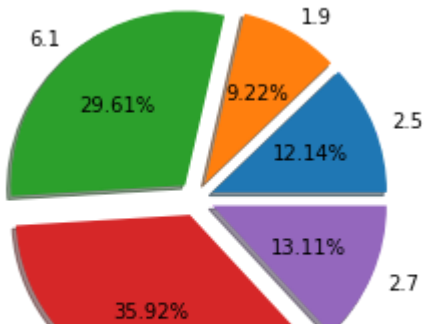

```
y = df_3["Scores"]
myexplode = [0.1,0.1,0.1,0.1,0.1]
mylabels =[62, 41, 42, 17, 95]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



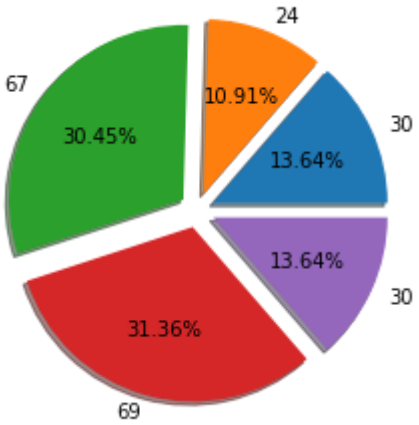

```
df_4= pd.DataFrame(df[16:21])
df_4
```

	Hours	Scores
16	2.5	30
17	1.9	24
18	6.1	67
19	7.4	69
20	2.7	30

```
y = df_4["Hours"]
myexplode = [0.1,0.1,0.1,0.1,0.1]
mylabels =[2.5, 1.9, 6.1, 7.4, 2.7]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



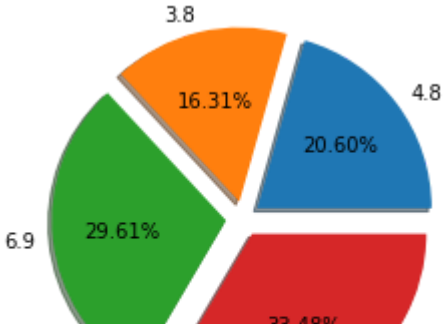

```
y = df_4["Scores"]
myexplode = [0.1,0.1,0.1,0.1,0.1]
mylabels =[30, 24, 67, 69, 30]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



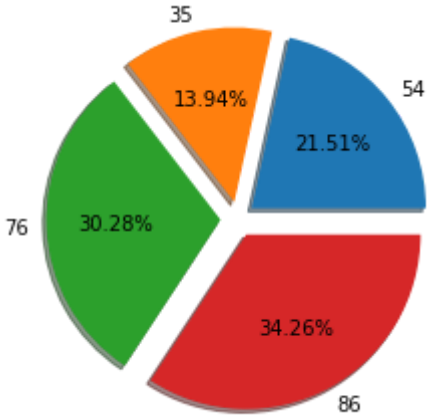

```
df_5 = pd.DataFrame(df[21:26])
df_5
```

	Hours	Scores
21	4.8	54
22	3.8	35
23	6.9	76
24	7.8	86

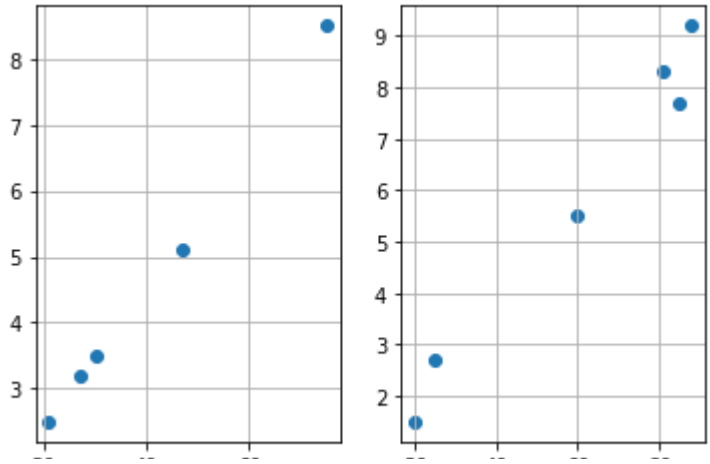
```
y = df_5["Hours"]
myexplode = [0.1,0.1,0.1,0.1]
mylabels =[4.8, 3.8, 6.9, 7.8]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



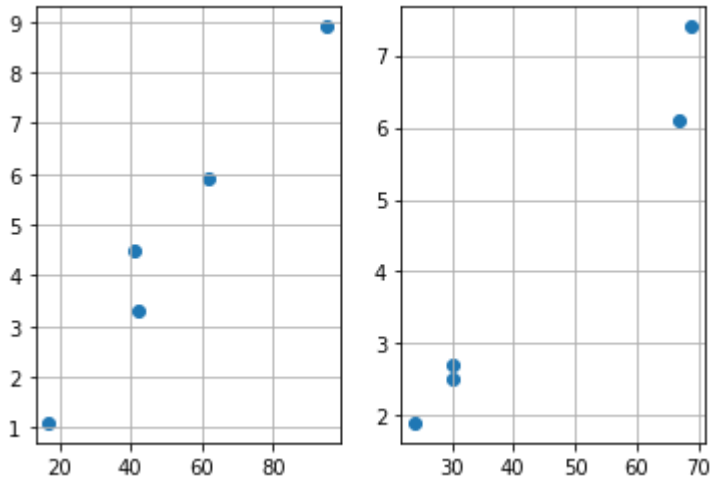

```
y = df_5["Scores"]
myexplode = [0.1,0.1,0.1,0.1]
mylabels =[54, 35, 76, 86]

plt.pie(y,labels=mylabels, explode = myexplode, shadow = True,autopct='%.2f%%')
plt.show()
```



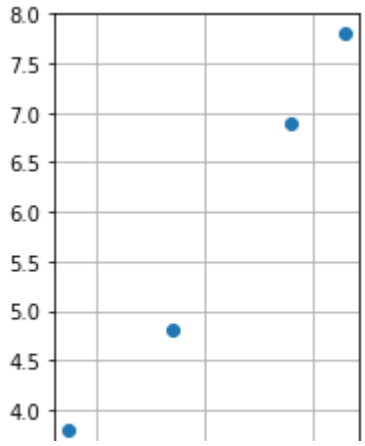

```
import matplotlib.pyplot as plt
import numpy as np

#plot 1:
x = df_1["Scores"]
y = df_1["Hours"]

plt.subplot(1, 2, 1)
plt.scatter(x,y)
plt.grid()

#plot 2:
x = df_2["Scores"]
y = df_2["Hours"]


plt.subplot(1, 2, 2)
plt.scatter(x,y)
plt.grid()


plt.show()
```



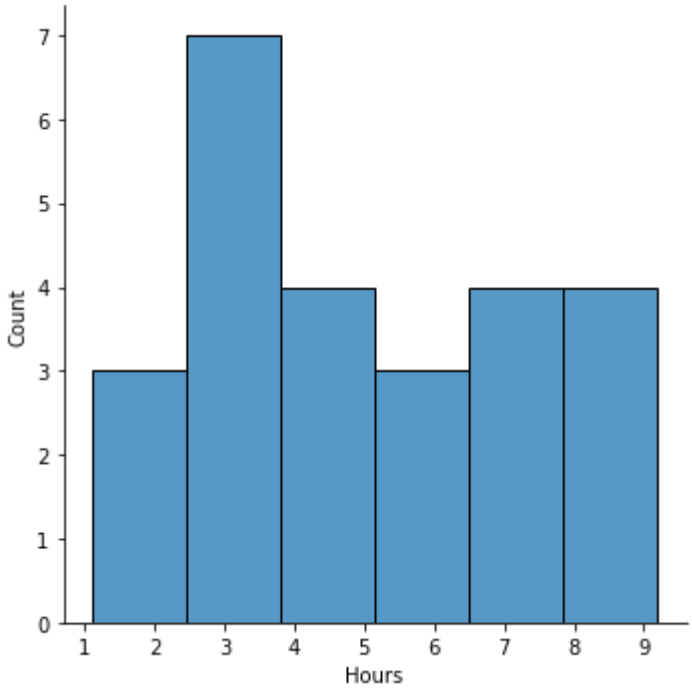
```
#plot 3:
x = df_3["Scores"]
y = df_3["Hours"]

plt.subplot(1, 2, 1)
plt.scatter(x,y)
plt.grid()

#plot 4:
x = df_4["Scores"]
y = df_4["Hours"]


plt.subplot(1, 2, 2)
plt.scatter(x,y)
plt.grid()#
```



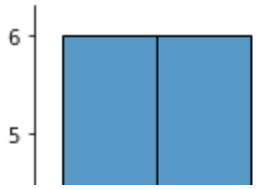
```
#plot 5:
x = df_5["Scores"]
y = df_5["Hours"]

plt.plot(1, 2, 1)
plt.scatter(x,y)
plt.grid()
```




```
import matplotlib.pyplot as plt
import seaborn as sns
sns.displot(df["Hours"])
plt.show()
```




```
sns.displot(df["Scores"])
plt.show()
```




```
## Percentiles
x = np.percentile(df["Scores"], 10)
print("Score 10 or below =",x)
x = np.percentile(df["Scores"], 20)
print("Score 20 or below =",x)
x = np.percentile(df["Scores"], 30)
print("Score 30 or below =",x)
x = np.percentile(df["Scores"], 40)
print("Score 40 or below =",x)
x = np.percentile(df["Scores"], 50)
print("Score 50 or below =",x)
x = np.percentile(df["Scores"], 60)
print("Score 60 or below =",x)
x = np.percentile(df["Scores"], 70)
print("Score 70 or below =",x)
x = np.percentile(df["Scores"], 80)
print("Score 80 or below =",x)
x = np.percentile(df["Scores"], 90)
print("Score 90 or below =",x)
x = np.percentile(df["Scores"], 100)
print("Score 100 or below =",x)
```

```
Score 10 or below = 22.200000000000003
Score 20 or below = 26.6
Score 30 or below = 30.0
Score 40 or below = 38.60000000000001
Score 50 or below = 47.0
Score 60 or below = 60.8
Score 70 or below = 68.6
Score 80 or below = 77.00000000000001
Score 90 or below = 85.6
Score 100 or below = 95.0
```

```
y = np.percentile(df["Hours"], 1)
print("Study 1 hour or below =",y)
y = np.percentile(df["Hours"], 2)
print("Study 2 hour or below =",y)
y = np.percentile(df["Hours"], 3)
print("Study 3 hour or below =",y)
y = np.percentile(df["Hours"], 4)
print("Study 4 hour or below =",y)
y = np.percentile(df["Hours"], 5)
print("Study 5 hour or below =",y)
y = np.percentile(df["Hours"], 6)
print("Study 6 hour or below =",y)
y = np.percentile(df["Hours"], 7)
print("Study 7 hour or below =",y)
y = np.percentile(df["Hours"], 8)
```

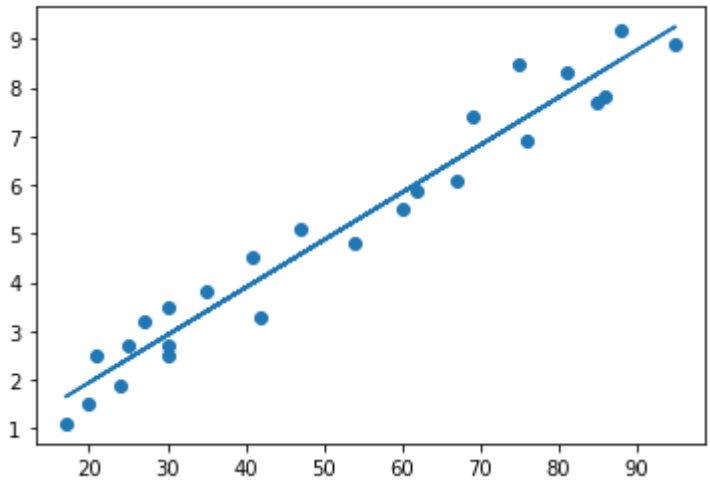
```
print("Study 8 hour or below =",y)
y = np.percentile(df["Hours"], 9)
print("Study 9 hour or below =",y)
y = np.percentile(df["Hours"], 10)
print("Study 10 hour or below =",y)
```

```
Study 1 hour or below = 1.1960000000000002
Study 2 hour or below = 1.292
Study 3 hour or below = 1.388
Study 4 hour or below = 1.484
Study 5 hour or below = 1.58
Study 6 hour or below = 1.676
Study 7 hour or below = 1.772
Study 8 hour or below = 1.8679999999999999
Study 9 hour or below = 1.996
Study 10 hour or below = 2.14
```

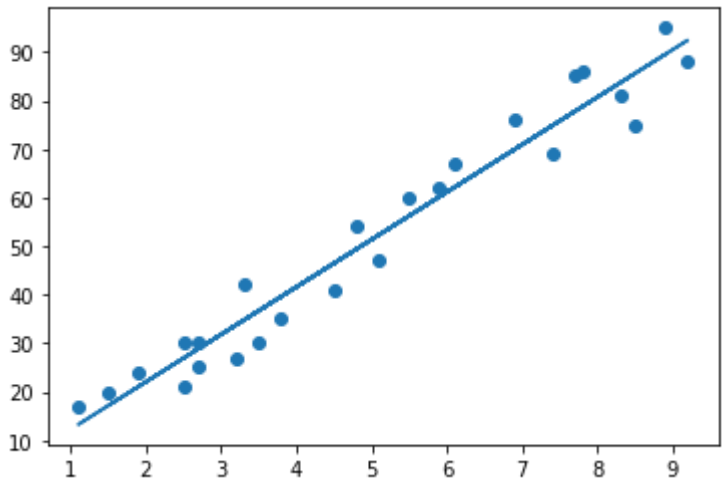
Linear Regression

Use of scipy library

```
import matplotlib.pyplot as plt
from scipy import stats

x = df["Scores"]
y = df["Hours"]

slope, intercept, r, p, std_err = stats.linregress(x, y)

def relation(x):
    return slope * x + intercept

linear = list(map(relation, x))

plt.scatter(x, y)
plt.plot(x, linear)
plt.show()
```



```
import matplotlib.pyplot as plt
from scipy import stats

x = df["Hours"]
y = df["Scores"]

slope, intercept, r, p, std_err = stats.linregress(x, y)

def relation(x):
    return slope * x + intercept

linear = list(map(relation, x))

plt.scatter(x, y)
plt.plot(x, linear)
plt.show()
```




```
X = df.iloc[:, :-1].values
y = df.iloc[:, 1].values
X_train, X_test, y_train, y_test = train_test_split(X, y,train_size=0.80,test_size=0.20,ra



from sklearn.linear_model import LinearRegression
linearRegressor= LinearRegression()
linearRegressor.fit(X_train, y_train)
y_predict= linearRegressor.predict(X_train)


regressor = LinearRegression()
regressor.fit(X_train, y_train)
print("Sucessfull")
```

```
     Sucessfull
```

```
print('Score of student who studied for 9.25 hours a dat', regressor.predict([[9.25]]))
```

```
     Score of student who studied for 9.25 hours a dat [92.38611528]
```


0s
completed at 1:29 AM