Name: Payal Shinde

Domain: DATA SCIENCE AND BUSINESS ANALYTICS

Task 1: PREDICTION USING SUPERVISED LEARNING

Language:Python

Dataset Link:http://bit.ly/w-data

In [28]:
```
#importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
from sklearn.linear_model import LinearRegression as lr
from sklearn.model_selection import train_test_split as tts
```

In [11]:
```
# reading the data
url='http://bit.ly/w-data'
data = pd.read_csv(url)
data.head(10)
```

Out[11]:

	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

In [10]:
```
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
```

In [12]:
```
#to check whether any duplicate value or missing value is present or not
data.isnull().sum()
```

Out[12]:
```
Hours     0
Scores    0
dtype: int64
```

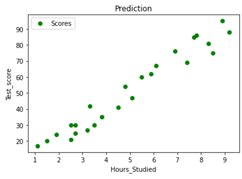
In [13]:
```
#analysis on data
data.describe()
```

Out[13]:

	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

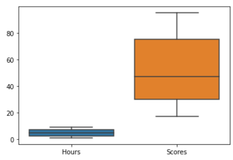
Plotting the dataset

In [14]:
```
data.plot(x='Hours',y='Scores', style='go')
plt.title('Prediction' )
plt.xlabel('Hours_Studied')
plt.ylabel('Test_score')
plt.show()
```



In [15]:
```
sns.boxplot(data=data[['Hours','Scores']])
```

Out[15]:
```
<AxesSubplot:>
```



Preparing Data

In [16]:
```
X = data.iloc[:,:-1].values
Y= data.iloc[:,1].values
```

In [39]:
```
X
```

Out[39]:
```
array([[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]])
```

In [17]:
```
Y
```

Out[17]:
```
array([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], dtype=int64)
```

Implementing Training Sets and Test Sets

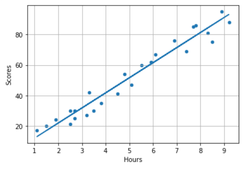
In [19]:
```
#we are going to use 20% of our data for testing and rest for training the dataset
X_train,x_test,Y_train,y_test = tts(X,Y,test_size=0.20,random_state = 0)
```

Training the Algorithm

In [20]:
```
#predicting the percantage of the marks
Reg = lr()
Reg.fit(X_train,Y_train)
```

Out[20]:
```
LinearRegression()
```

In [26]:
```
L = Reg.coef_*X+ Reg.intercept_
data.plot.scatter(x='Hours', y='Scores')
plt.plot(X,L)
plt.grid()
plt.show()
```



In [88]:
```
y_pred=Reg.predict(x_test)
print(y_pred)
```

```
[16.88414476 33.73226078 75.357018   26.79480124 60.49103328]
```

In [98]:
```
#final prediction for the case that if a student studies 9.25 hrs/day

h = np.array([[9.25]])
p = Reg.predict(h)
print('No of hours =', h[0][0])
print('Predicted Score =',p[0])
```

```
No of hours = 9.25
Predicted Score = 93.69173248737539
```

Evaluating the model

In [101]:
```
print('Mean Absolute Error = ', metrics.mean_absolute_error(y_test,y_pred))
```

```
Mean Absolute Error =  4.183859899002982
```