Name :- Manni Deol

Email :- netdeol2000@gmail.com

Task 1 : Prediction using Supervised Machine Learning

GRIP @ The Sparks Foundation

In this regression task I tried to predict the percentage of marks that a student is expected to score based upon the number of hours they studied.

This is a simple linear regression task as it involves just two variables.

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

Data Load

In [2]:

```
#data = pd.read_csv('student_info.csv')
# Reading data from remote link
url = "http://bit.ly/w-data"
data = pd.read_csv(url)
print("Data imported successfully")

data.head(10)
```

```
Data imported successfully
```

Out[2]:

	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 [3]:

```
data.tail()
```

Out[3]:

	Hours	Scores

	Hours	Scores
20	2.7	30
21	4.8	54
22	3.8	35
23	6.9	76
24	7.8	86

In [4]:

```
data.isnull().sum().head()
# True = null
# False = Not NUll
```

Out[4]:

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

In [5]:

```
data.shape
```

Out[5]:

```
(25, 2)
```

Data Discover and Visualization

In [6]:

```
data.info()
```

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

In [7]:

```
data.describe()
```

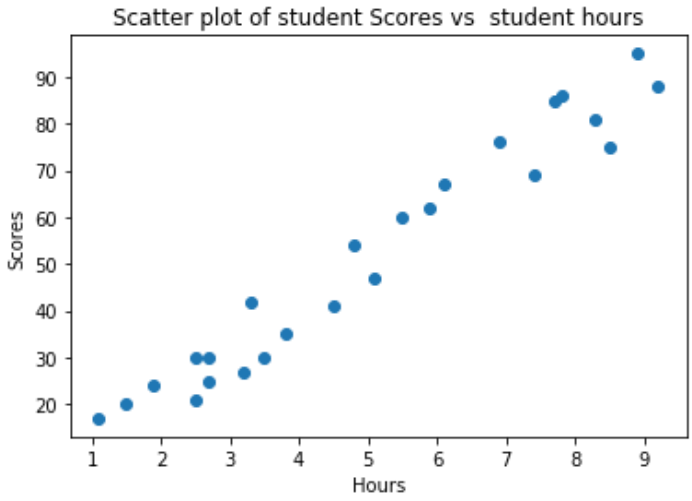
Out[7]:

	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 [8]:

```
plt.scatter(x = data.Hours , y = data.Scores)
plt.xlabel('Hours')
plt.ylabel('Scores')
plt.title('Scatter plot of student Scores vs  student hours')
```

```
plt.show()
```

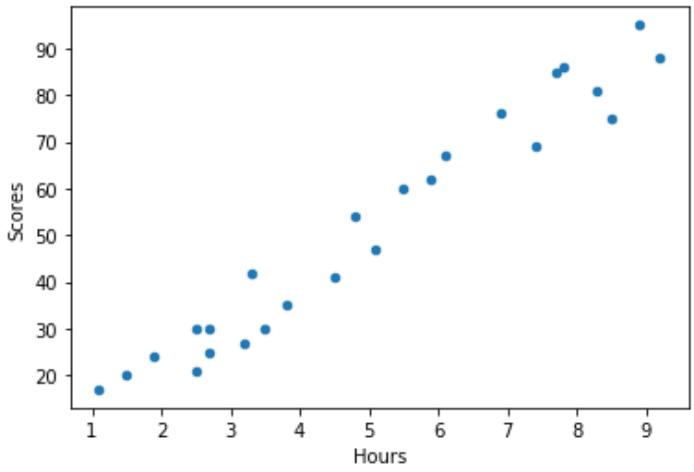


In [9]:

```
#Another way to plot graph
data.plot(kind='scatter', x='Hours', y='Scores',alpha=1)
```

Out[9]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d12b180448>
```

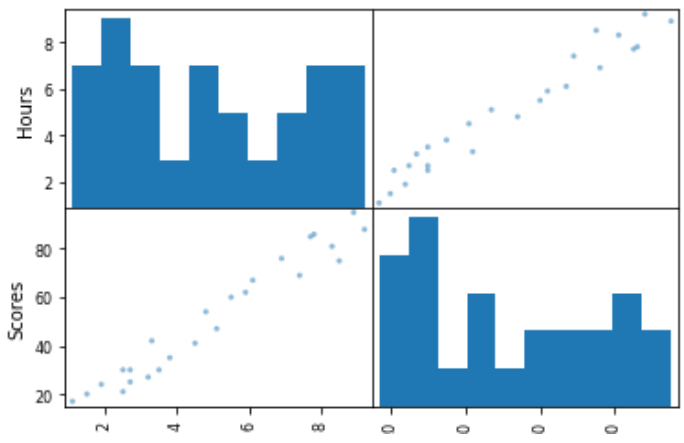


In [10]:

```
from pandas.plotting import scatter_matrix
attributes = ['Hours' , 'Scores']
scatter_matrix(data[attributes] , )
```

Out[10]:

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000001D12D493788>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001D12D4B6848>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000001D12D4EDCC8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001D12D525708>]],
      dtype=object)
```



Fill Missing Attributes

In [11]:

```
data.mean()
data1 = data.fillna(data.mean())
#data1.describe()
data1.isnull().sum()
```

Out[11]:

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

looking for correlation

In [12]:

```
corr_matrix = data.corr()
corr_matrix['Scores'].sort_values(ascending=False)
```

Out[12]:

```
Scores    1.000000
Hours     0.976191
Name: Scores, dtype: float64
```

Train Test Split

In [13]:

```
X = data1.drop('Scores' , axis = "columns")
y = data1.drop('Hours' , axis ='columns')
print(f"shape of x is {X.shape} \nShape of y is {y.shape}")
```

```
shape of x is (25, 1)
Shape of y is (25, 1)
```

In [14]:

```
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 =
51)
print('x_train = ',len(x_train))
print('x_test = ',len(x_test))
print('y_train = ',len(y_train))
print('y_test = ',len(y_test))
```

```
x_train =  20
x_test =  5
y_train =  20
y_test =  5
```

Build a Machine Learning Model

In [15]:

```
# y = m*x+c
from sklearn.linear_model import LinearRegression
lr_model = LinearRegression()
```

In [16]:

```
lr_model.fit(x_train, y_train)
x_pre = lr_model.predict(x_test)
lr_model.score(x_test,y_test)
```

Out[16]:

```
0.9238518102278781
```

In [17]:

```
dataframe = pd.DataFrame(np.c_[x_test, y_test,x_pre], columns = ['study_hours','Score','
Score_predicted'])
dataframe
```

Out[17]:

	study_hours	Score	Score_predicted
0	5.5	60.0	55.305827
1	7.7	85.0	76.347369
2	6.9	76.0	68.695899
3	8.3	81.0	82.085971
4	2.7	30.0	28.525682

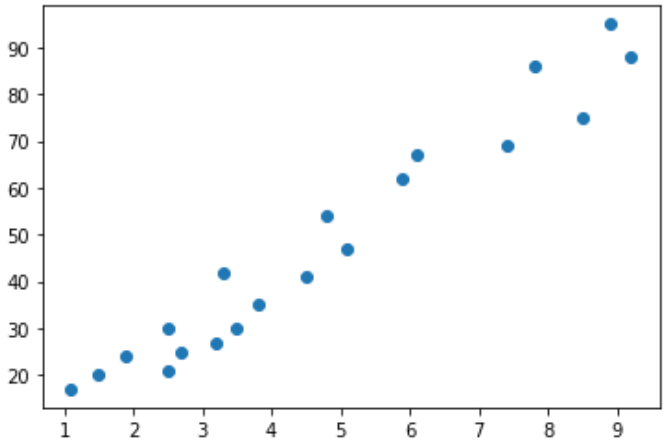
Fine Tune your Model

In [18]:

```
plt.scatter(x_train,y_train)
```

Out[18]:

```
<matplotlib.collections.PathCollection at 0x1d13de38e88>
```

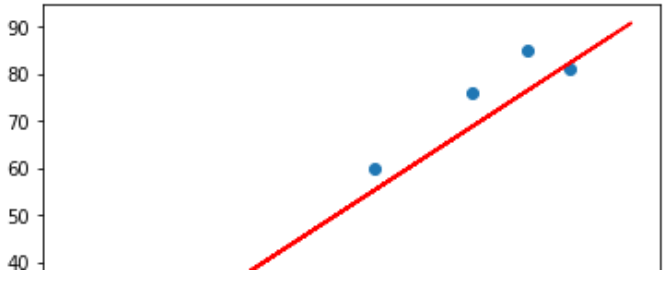


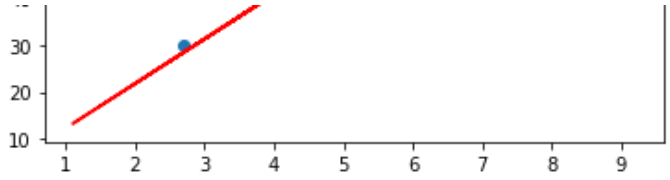
In [19]:

```
plt.scatter(x_test, y_test)
plt.plot(x_train, lr_model.predict(x_train), color = 'r')
```

Out[19]:

```
[<matplotlib.lines.Line2D at 0x1d13de42688>]
```





Save your Model

In [20]:

```
import joblib
joblib.dump(lr_model ,'Student_marks_predictor_model.pkl')
```

Out[20]:

```
['Student_marks_predictor_model.pkl']
```

In [21]:

```
model = joblib.load('Student_marks_predictor_model.pkl')
model
```

Out[21]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [22]:

```
model.predict([[5]])
```

Out[22]:

```
array([[50.52365786]])
```

Task 1 is Complete

Conclusion

I was successfully able to carry-out Prediction using Supervised ML task and was able to evaluate the model's performance on various parameters.

Thank You

In [ ]:

```

```