# The Sparks Foundation Data Science & Business Analytics Intern Task 1: Prediction using Supervised ML For this task, we will be applying Linear Regression for predicting student's percentage based on the no. of study hours Data Source:http://bit.ly/w-data Problem statement: What will be the predicted score if student studies for 9.25 hours/day.

In [1]:

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

In [2]:

```
df = pd.read_excel("C:\\Users\\Pratima Dhar\\Downloads\\student score.xlsx")
df.head()
```

Out[2]:

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

In [3]:

```
df.shape
```

Out[3]:

```
(25, 2)
```

In [4]:

```
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
```

In [5]:

```
df.describe()
```

Out[5]:

|  | 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 [6]:

```
num_col=df.select_dtypes(include=np.number).columns
print("numerical columns: \n",num_col)
```

```
numerical columns:
 Index(['Hours', 'Scores'], dtype='object')
```

In [7]:

```
df.corr()
```

Out[7]:

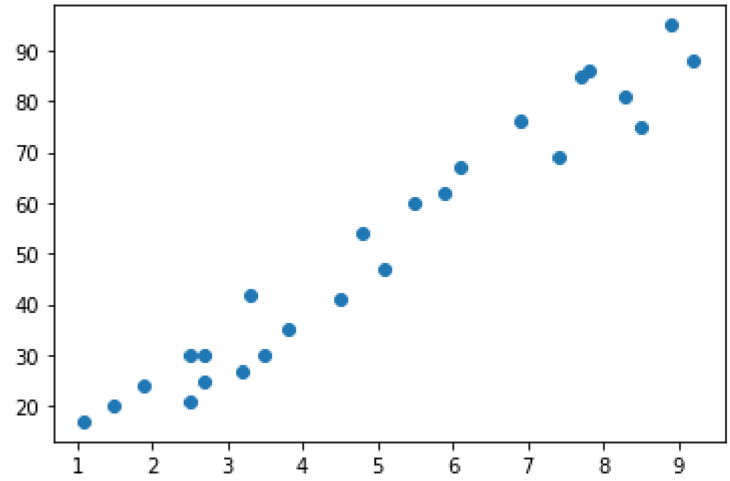
|  | Hours | Scores |
|---|---|---|
| **Hours** | 1.000000 | 0.976191 |
| **Scores** | 0.976191 | 1.000000 |

In [8]:

```
x='Hours'
y='Scores'
```

In [9]:

```
plt.scatter(x='Hours',y='Scores',data=df)
plt.show()
```



we can clearly observe There is positive correlation between hours and marks, these two variables can make a straight trend line, making them ideal for Linear Regression application.

In [10]:

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn import linear_model
```

In [11]:

```
x1 = df.iloc[:,0].values
y1 = df.iloc[:,1].values

x = x1.reshape(-1,1)
y = y1.reshape(-1,1)
```

We make use of Scikit Learn's train_test_split() method for splitting data into training and testing data in a 70:30 split ratio.

In [12]:

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3)
```

In [13]:

```
from sklearn.linear_model import LinearRegression
linreg = LinearRegression()
model=linreg.fit(x_train, y_train)
model
```

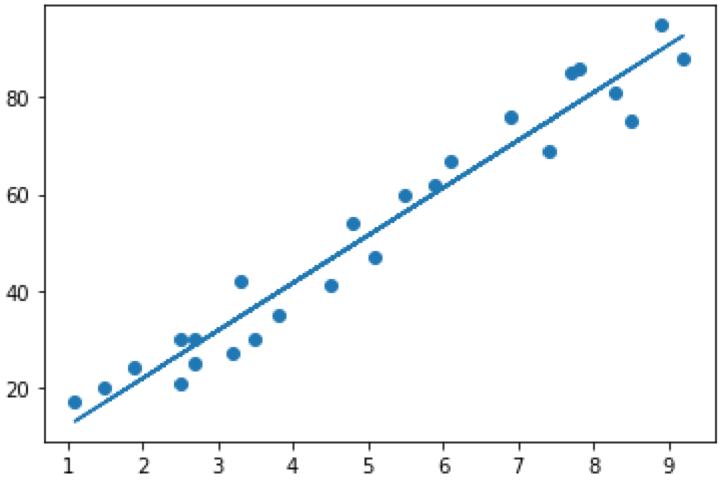
Out[13]:

```
LinearRegression()
```

In [14]:

```
#plottting regression line
line = linreg.coef_*x+linreg.intercept_

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



In [15]:

```
predictions =linreg.predict(x_test)
predictions
```

Out[15]:

```
array([[28.72095546],
       [85.89439035],
       [28.72095546],
       [16.89196892],
       [78.99414821],
       [20.83496444],
       [70.12240831],
       [36.60694648]])
```

In [16]:

```
#model evaulation
from sklearn.metrics import mean_absolute_error
print("MSE:", mean_absolute_error(y_test, predictions))
```

```
MSE: 5.207230868183423
```

In [17]:

```
Hours = np.array(9.25)
Hours = Hours.reshape(-1, 1)
pred = linreg.predict(Hours)
print("If the student studies for 9.25 hours, he is expected to score {}.".format(pred
))
```

```
If the student studies for 9.25 hours, he is expected to score [[93.287506
94]].
```

student studies for 9.25 hours/day, he is expected to score 91.47 marks.

In [ ]: