Predict-Percentage-Marks-Vs-Hours-Studied

Importing Libraries

In [78]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
get_ipython().run_line_magic('matplotlib', 'inline')
```

Import CSV file

In [79]:

```
dataset = pd.read_csv('D:student_scores.csv')
```

In [80]:

```
dataset
```

Out[80]:

	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

	Hours	Scores
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
23	6.9	76
24	7.8	86

Explore the dataset

In [81]:

```
dataset.shape
```

Out[81]:

```
(25, 2)
```

In [82]:

```
dataset.head()
```

Out[82]:

	Hours	Scores
0	2.5	21
1	5.1	47

	Hours	Scores
2	3.2	27
3	8.5	75
4	3.5	30

In [83]:

```
dataset.describe()
```

Out[83]:

	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

check data types for each column

In [84]:

```
print (dataset.dtypes)
```

```
Hours     float64
Scores      int64
dtype: object
```

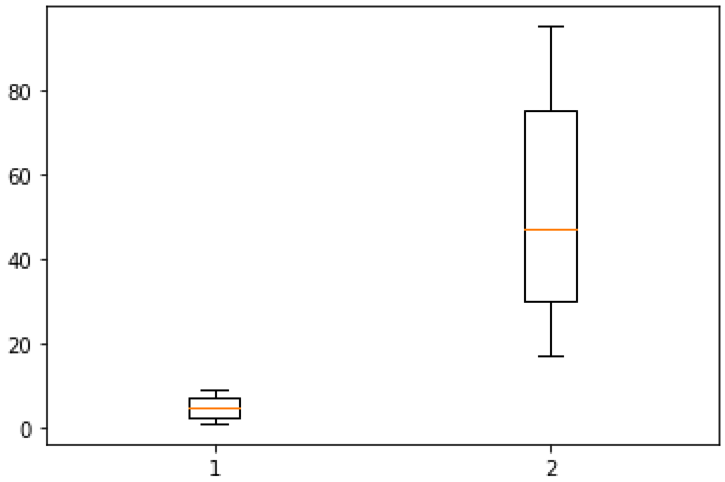
Check for outlier values

In [85]:

```
plt.boxplot(dataset.values)
```

Out[85]:

```
{'whiskers': [<matplotlib.lines.Line2D at 0x27844d37a90>,
  <matplotlib.lines.Line2D at 0x27844d4d880>,
  <matplotlib.lines.Line2D at 0x27844d688e0>,
  <matplotlib.lines.Line2D at 0x27844d59d60>],
 'caps': [<matplotlib.lines.Line2D at 0x27844d4d160>,
  <matplotlib.lines.Line2D at 0x27844d4db80>,
  <matplotlib.lines.Line2D at 0x27844d599d0>,
  <matplotlib.lines.Line2D at 0x27844d59e50>],
 'boxes': [<matplotlib.lines.Line2D at 0x27844d37790>,
  <matplotlib.lines.Line2D at 0x27844d68b20>],
 'medians': [<matplotlib.lines.Line2D at 0x27844d4d3a0>,
  <matplotlib.lines.Line2D at 0x27844d3d7c0>],
 'fliers': [<matplotlib.lines.Line2D at 0x27844d4d3d0>,
  <matplotlib.lines.Line2D at 0x27844d3d5e0>],
 'means': []}
```



dataset.plot(x='Hours',y= 'Scores',style='o') plt.title('Hours vs Percentage') plt.xlabel('Hours Studied') plt.ylabel('Percentage Score') plt.show()

Prepare the data to train the model

In [86]:

```
X= dataset.iloc[:, :-1].values
```

```
y= dataset.iloc[:, 1].values
```

```
In [87]: 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,rando
         m_state=0)
```

```
In [88]: from sklearn.linear_model import LinearRegression
         regressor = LinearRegression()
         regressor.fit(X_train, y_train)

Out[88]: LinearRegression()
```

To see the value of the intercept and slop calculated by the linear regression algorithm for our dataset

```
In [89]: print(regressor.intercept_)

         2.018160041434683
```

```
In [90]: print(regressor.coef_)

         [9.91065648]
```

```
In [91]: y_pred = regressor.predict(X_test)
```

```
In [92]: df = pd.DataFrame({'Actual':y_test,'Predicted': y_pred})
         df

Out[92]:
```

	Actual	Predicted
0	20	16.884145
1	27	33.732261
2	69	75.357018
3	30	26.794801

	Actual	Predicted
4	62	60.491033

Evaluate the algorithm by 1.MAE 2.MSE 3.RMSE

In [93]:

```
from sklearn import metrics
print('Mean Absolute Error:',metrics.mean_absolute_error(y_test,y_pred
))
print('Mean Squared Error:',metrics.mean_squared_error(y_test,y_pred))
print('Root Mean Squared Error:',np.sqrt(metrics.mean_squared_error(y_t
est,y_pred)))
```

```
Mean Absolute Error: 4.183859899002975
Mean Squared Error: 21.5987693072174
Root Mean Squared Error: 4.6474476121003665
```

In [94]:

```
hours = float(input(' Enter the hours of study:- '))
predicted = regressor.predict([[hours]])
print(" If a student studies for = {}".format(hours),"hours, then his/h
er predicted score is = {}%".format(predicted))
```

```
 Enter the hours of study:- 9
 If a student studies for = 9.0 hours, then his/her predicted score is
= [91.21406837]%
```