jupyter linear regression on percentage of student based on their study h... Last Checkpoint: Yesterday at 12:02 PM (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Pytho

Run Code

In [2]:
```
# importing all libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
df=pd.read_csv("https://raw.githubusercontent.com/AdiPersonalWorks/Random/master/student_scores%20-%20student_scores.csv")
print ("data imported sucessfully")

df.head(10)
```

```
data imported sucessfully
```

Out[3]:

	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

jupyter linear regression on percentage of student based on their study h... Last Checkpoint: Yesterday at 12:02 PM (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted

Run Code

In [4]:
```
df.shape
```

Out[4]: (25, 2)

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

jupyter linear regression on percentage of student based on their study h... Last Checkpoint: Yesterday at 12:02

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [6]:
```
df.info()
```

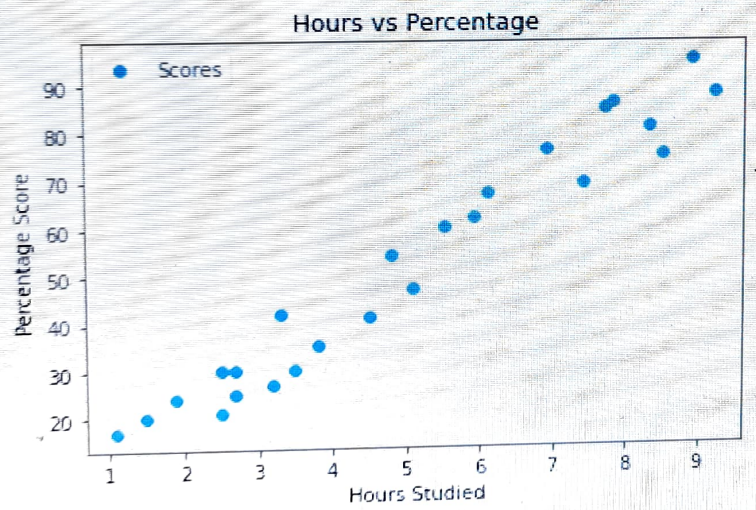
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 528.0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

jupyter linear regression on percentage of student based on their study h... Last Checkpoint: Yesterday at 12:02 PM (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted

Run Code

In [10]:
```
df.plot(x='Hours',y='Scores',style='o')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.title('Hours vs Percentage')
plt.show()
```



jupyter linear regression on percentage of student based on their study h... Last Checkpoint: Yesterday at 12:02 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Code

In [11]:
```
df.corr()
```

Out[11]:

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

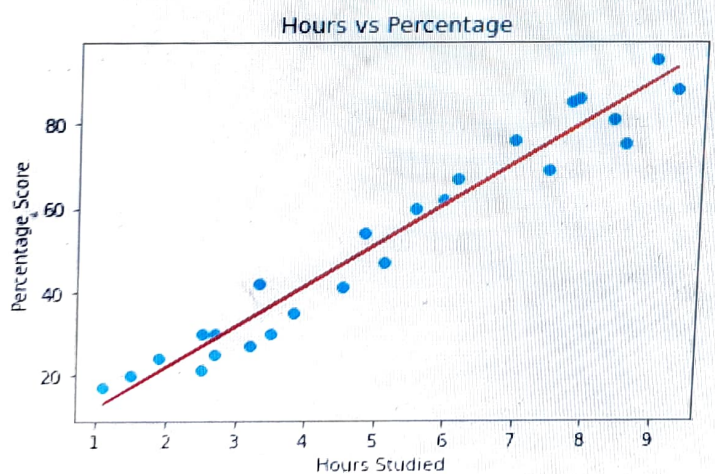
In [12]:
```
x=df.iloc[:,:-1].values
y=df.iloc[:,1].values
```

In [13]:
```
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=0)
```

In [14]:
```
from sklearn.linear_model import LinearRegression
regressor=LinearRegression()
regressor.fit(x_train,y_train)
print("Score:",regressor.score(x_train,y_train))
print("Training complete.")
```

```
Score: 0.9515510725211552
Training complete.
```

In [16]:
```
m=regressor.coef_
c=regressor.intercept_
# equation of line
line=m*x+c
# plotting for the test data
plt.scatter(x,y)
plt.plot(x,line,"r")
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.title('Hours vs Percentage')
plt.show()
```



jupyter linear regression on percentage of student based on their study h... Last Checkpoint: Yesterday at 12:02 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Code

In [17]:
```
print("Testing data-In Hours\n",x_test)
# predicting the scores
y_pred=regressor.predict(x_test)
```

```
Testing data-In Hours
 [[1.5]
 [3.2]
 [7.4]
 [2.5]
 [5.9]]
```

In [18]:
```
print("Predicted y:\n",y_pred)
```

```
Predicted y:
 [16.88414476 33.73226078 75.357018   26.79480124 60.49103328]
```

In [20]:
```
df=pd.DataFrame({'Actual':y_test,'Predicted':y_pred})
df
```

Out[20]:

	Actual	Predicted
0	20	16.884145
1	27	33.732261
2	69	75.357018
3	30	26.794801
4	62	60.491033

3	30	26.794801
4	62	60.491033

In [21]:
```
# test can be done with our own data
hours=9.25
own_pred=regressor.predict([[hours],])
print("No of Hours={}".format(hours))
print("Predicted Score={}".format(own_pred[0]))
```

```
No of Hours=9.25
Predicted Score=93.69173248737538
```

In [22]:
```
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('Accuracy:',regressor.score(x_test,y_test)*100,"%")
```

```
Mean Absolute Error: 4.183859899002975
Mean Squared Error: 21.5987693072174
Accuracy: 94.54906892105356 %
```

In [ ]: