GRIP-THE SPARKS FOUNDATION DATA SCIENCE AND BUSINESS ANALYTICS AUTHOR: YASHIKA TASK1 : PREDICTIONUSING SURPERVISED ML

This is the task to predict the percentage of marks of the student based on the number of hours they studied. this is a simple linear regression task involving two features.

In [33]:

```
#importing required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [4]:

```
#Reading the data
df=pd.read_csv("D:\data set.csv")
```

In [3]:

```
df
```

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
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

	Hours	Scores
24	7.8	86

In [4]:

```
df.shape
```

Out[4]:

```
(25, 2)
```

In [5]:

```
df.describe()
```

Out[5]:

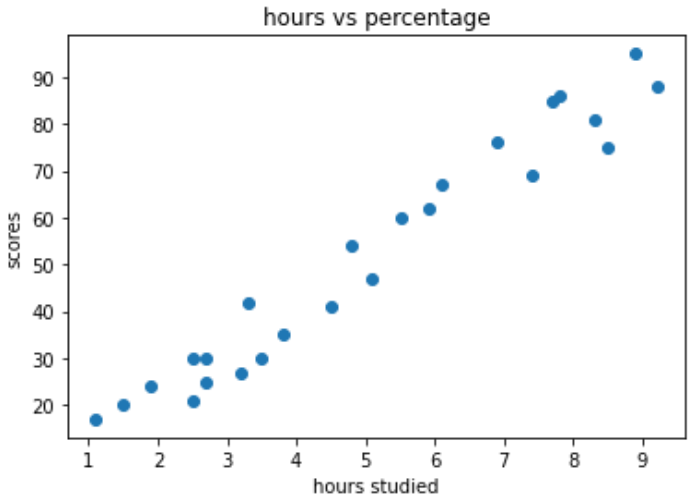
	Hours	Scores
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

In [5]:

```
plt.scatter(df['Hours'], df['Scores'])
plt.title("hours vs percentage")
plt.xlabel("hours studied")
plt.ylabel("scores")
```

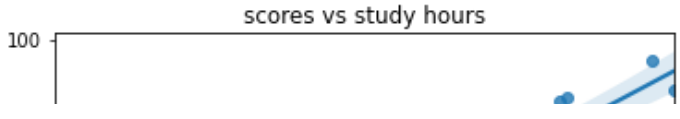
Out[5]:

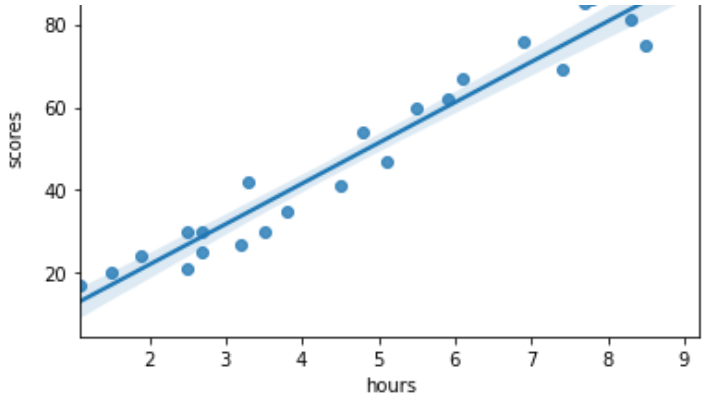
```
Text(0, 0.5, 'scores')
```



In [36]:

```
sns.regplot(x=df['Hours'],y=df['Scores'])
plt.title('scores vs study hours')
plt.ylabel('scores')
plt.xlabel('hours')
plt.show()
```





In [37]:

```
print(df.corr())
```

```
            Hours    Scores
Hours    1.000000  0.976191
Scores   0.976191  1.000000
```

In [6]:

```
#Training the model

#splitting the data

#defining x and y from the data.
x = df.iloc[:, :-1].values
y = df.iloc[:, 1].values
```

In [7]:

```
#spliting the data in two

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 [8]:

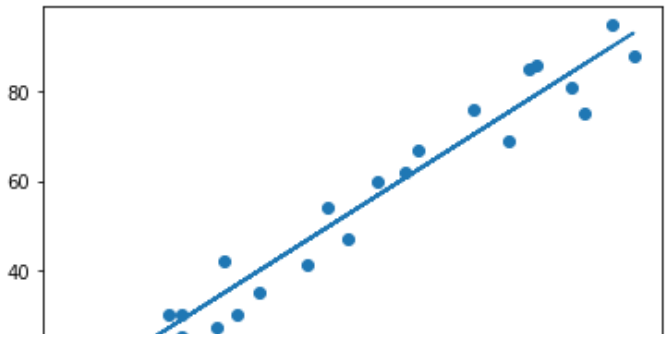
```
#fitting the data into the model

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

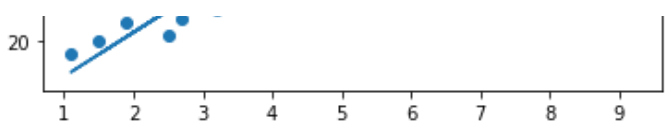
Out[8]:

```
LinearRegression()
```

In [9]:

```
m=reg.coef_
c=reg.intercept_
line=m*x+c
plt.scatter(x,y)
plt.plot(x,line)
plt.show()
```





In [38]:

```
#predicting scores

y_pred=reg.predict(x_test)
y_pred
```

Out[38]:

```
array([16.88414476, 33.73226078, 75.357018  , 26.79480124, 60.49103328])
```

In [39]:

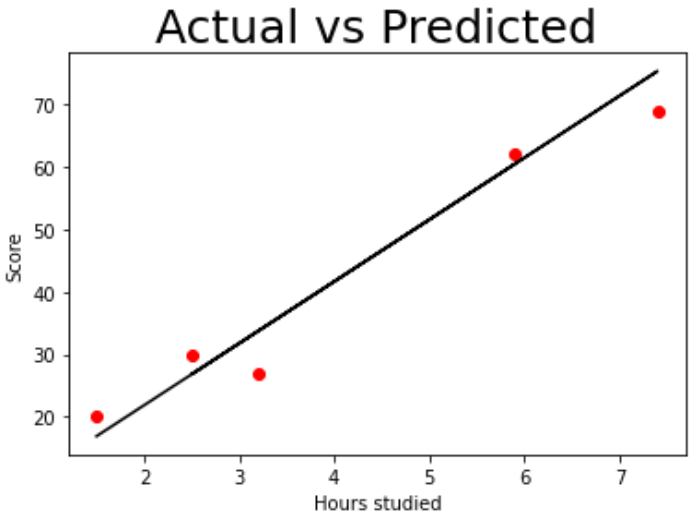
```
#comparing the predicted marks with the actual marks

actual_predicted=pd.DataFrame({'Actual':y_test,'predicted':y_pred})
actual_predicted
```

Out[39]:

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

In [26]:

```
#plotting predicted marks with actual marks

plt.scatter(x=x_test,y=y_test,color='red')
plt.plot(x_test,y_pred,color='black')
plt.title('Actual vs Predicted',size=25)
plt.xlabel('Hours studied')
plt.ylabel('Score')
plt.show()
```



In [25]:

```
#Model evaluation
#calculating the accuracy of the model

from sklearn.metrics import mean_absolute_error
print('mean absolute error:', mean_absolute_error(y_test,y_pred))
```

```
mean absolute error: 4.183859899002975
```

In [32]:

```
# predicted scores if he/she studied for 9.25 hours/day?

h=9.25
s=reg.predict([[h]])
print('if a student studies for {} hours per day he/she will score {} % in exams'.format
(h,s))
```

```
if a student studies for 9.25 hours per day he/she will score [93.69173249] % in exams
```