SIMPLE LINEAR REGRESSION

Our task is to predict the student score based on the number hours that he had spent on study. We had used simple linear regression which is the supervised regression technique, to predict the score of the student

In [1]:

```
#step 1: import all required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error,r2_score
```

```
c:\users\vamsi2001\appdata\local\programs\python\python39\lib\site-packages\num
py\_distributor_init.py:30: UserWarning: loaded more than 1 DLL from .libs:
c:\users\vamsi2001\appdata\local\programs\python\python39\lib\site-packages\num
py\.libs\libopenblas.EL2C6PLE4ZYW3ECEVIV3OXXGRN2NRFM2.gfortran-win_amd64.dll
c:\users\vamsi2001\appdata\local\programs\python\python39\lib\site-packages\num
py\.libs\libopenblas.XWYDX2IKJW2NMTWSFYNGFUWKQU3LYTCZ.gfortran-win_amd64.dll
  warnings.warn("loaded more than 1 DLL from .libs:"
```

In [2]:

```
#step 2: load the data
data=pd.read_csv('marks.csv')
data.head()
```

Out[2]:

	Hours	Scores
0	3.75	67.79
1	9.51	99.06
2	7.32	86.43
3	5.99	74.11
4	1.56	63.51

In [3]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   149 non-null    float64
 1   Scores  148 non-null    float64
dtypes: float64(2)
memory usage: 2.5 KB
```

In [5]:

```
data.isnull().sum()
```

Out[5]:

```
Hours     1
Scores    2
dtype: int64
```

In [6]:

```
# Handling Missing values
data['Hours']=data['Hours'].fillna(data['Hours'].mean())
data['Scores']=data['Scores'].fillna(data['Scores'].median())
```

In [7]:

```
data.isnull().sum()
```

Out[7]:

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

In [8]:

```
data.info()
```

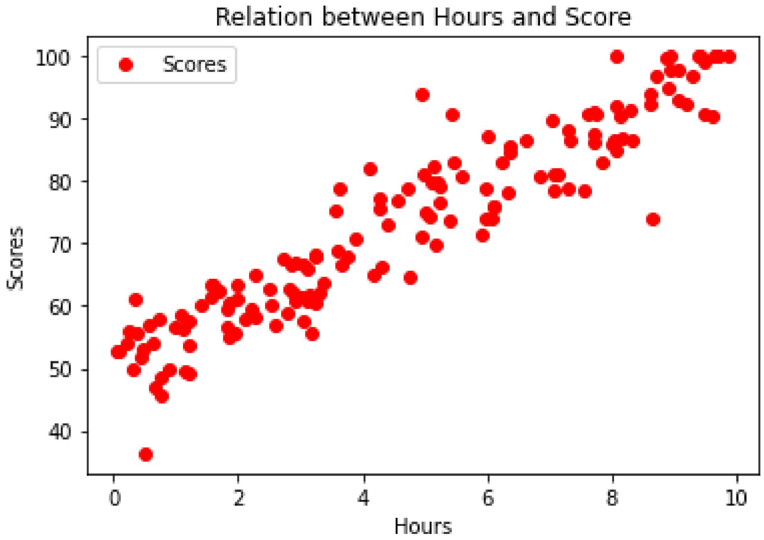
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   150 non-null    float64
 1   Scores  150 non-null    float64
dtypes: float64(2)
memory usage: 2.5 KB
```

In [9]:

```
#step 3:plot the relationship between features(columns)
data.plot(x='Hours',y='Scores',style='o',color='red')
plt.title('Relation between Hours and Score')
plt.xlabel('Hours')
plt.ylabel('Scores')
```

Out[9]:

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



In [10]:

```
#step 4: split dataset
X=data.iloc[:,:-1].values
y=data.iloc[:,1].values
#X=np.array(data['Hours']).reshape(-1,1)
#y=np.array(data['Scores'])
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2)
```

In [11]:

```
#step 5: build the model
reg=LinearRegression()
reg.fit(X_train,y_train)
```

Out[11]:

```
LinearRegression()
```

In [12]:

```
#step 6:Predictions
y_pred=reg.predict(X_test)
y_pred
```

Out[12]:

```
array([77.05107029, 86.08942867, 84.86802889, 63.95766464, 56.18956204,
       94.05295524, 76.85564633, 55.99413807, 85.16116484, 65.8630483 ,
       87.26197246, 61.51486508, 53.50248252, 65.03249645, 74.90140668,
       75.19454263, 80.76412563, 59.36520147, 73.38687095, 63.56681671,
       97.52173062, 70.35779949, 89.50934805, 92.39185154, 52.13451477,
       66.84016813, 90.97502779, 85.99171669, 64.25080059, 75.29225461])
```

In [13]:

```
print("m value",reg.coef_)#m value
print("Intercept value",reg.intercept_)# intercept
```

```
m value [4.88559912]
Intercept value 50.3268430917942
```

In [14]:

```
#step 7:Model Evaluation
print("mean absolute error is",mean_absolute_error(y_test,y_pred))
print("r2 value",r2_score(y_test,y_pred))
```

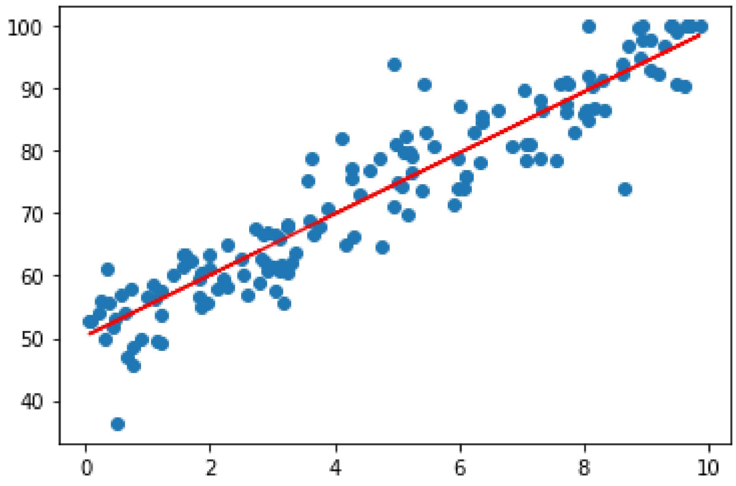
```
mean absolute error is 4.67946019193863
r2 value 0.8561505991497251
```

In [15]:

```
#step 8:plot Regression line
line=reg.coef_*X+reg.intercept_ #y=mx+c
plt.scatter(X,y)
plt.plot(X,line,c='red')
```

Out[15]:

```
[<matplotlib.lines.Line2D at 0x1e562da75b0>]
```



In [16]:

```
#step 9:predict using custom input
hr=np.array(8)# hours
pred=reg.predict(hr.reshape(-1,1))
print(pred)
```

```
[89.41163607]
```

In [ ]:

```

```

In [ ]:

```

```