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

Read the data

```
df=pd.read_csv('height-weight.csv')
df.head()
```

now visualize the data

```
plt.scatter(df['Weight'],df['Height'])
plt.xlabel("Weight")
plt.ylabel("Height")
```

now you can clearly visualize that there is linear relationship in the datasert

```
df.info()
```

```
df.describe()
```

find out there is any null value present in the dataset

```
df.isnull().sum()
```

divide dataset into dependent and independent feature

```
## divvide our dataset into independent and dependent features
X=df[['Weight']] ## idnependent feature
y=df['Height'] ##dependent features
```

now see the shape of the dataset

```
X.shape,y.shape
```

Now Import the traintestsplit

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test
=train_test_split(X,y,test_size=0.20,random_state=42)
```

Now find out the train and test dataset shape

```
X_train.shape,X_test.shape
y_train.shape,y_test.shape
```

now standardise the dataset with standard scaller function

```
from sklearn.preprocessing import StandardScaler
```

```
scaler=StandardScaler()
```

```
X_train=scaler.fit_transform(X_train)
```

```
X_test=scaler.transform(X_test)
```

Now check the model train dataset

```
X_test
```

Now visualize the train dataset

```
plt.scatter(X_train,y_train)
```

```
scaler.transform([[80]])
```

```
from sklearn.linear_model import LinearRegression
```

create object of the regression

```
regressor=LinearRegression()
```

```
## Training the train data
regressor.fit(X_train,y_train)
```

find the intecept of the train dataset

```
regressor.intercept_
```

now find the coefficient of the train datset

```
regressor.coef_
```

now visualize the best fit line and predicted the value

```
plt.scatter(X_train,y_train)
plt.plot(X_train,regressor.predict(X_train),'r')

### prediction of train data

1. predicted height output= intercept +coef_(Weights)
2. y_pred_train =157.5 + 17.03(X_train)

### prediction of test data
1. predicted height output= intercept +coef_(Weights)
2. y_pred_test =157.5 + 17.03(X_test)
```

```
## Prediction for test data
y_pred_test=regressor.predict(X_test)
```

```
y_pred_test
```

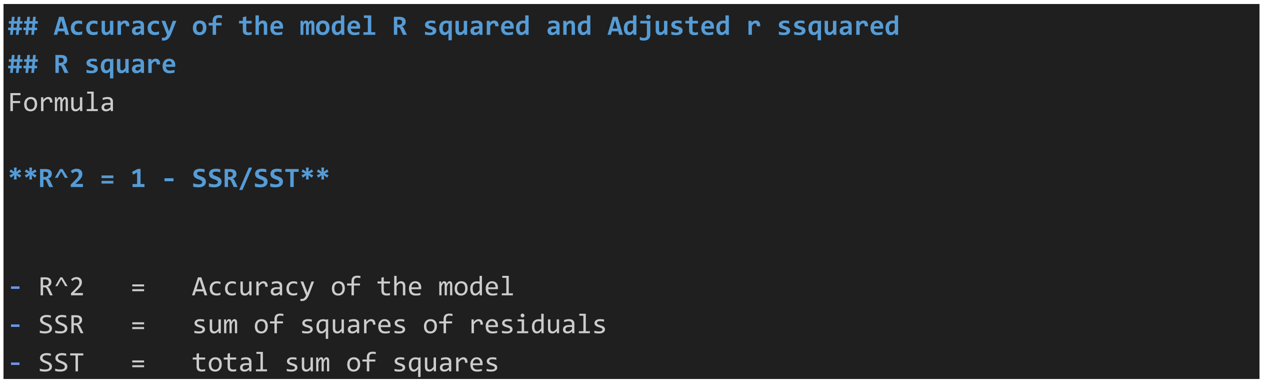
```
y_test
```

now check the

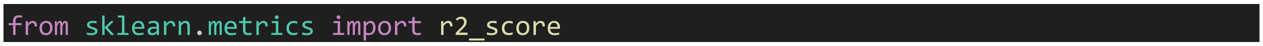
```
## Performance Metrics MAE,MSE,RMSE
```

```
from sklearn.metrics import mean_squared_error,mean_absolute_error
```

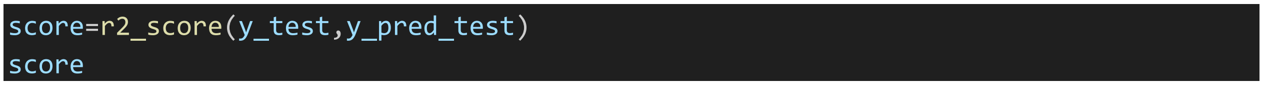
```
mse=mean_squared_error(y_test,y_pred_test)
mae=mean_absolute_error(y_test,y_pred_test)
rmse=np.sqrt(mse)
print(mse)
print(mae)
print(rmse)
```



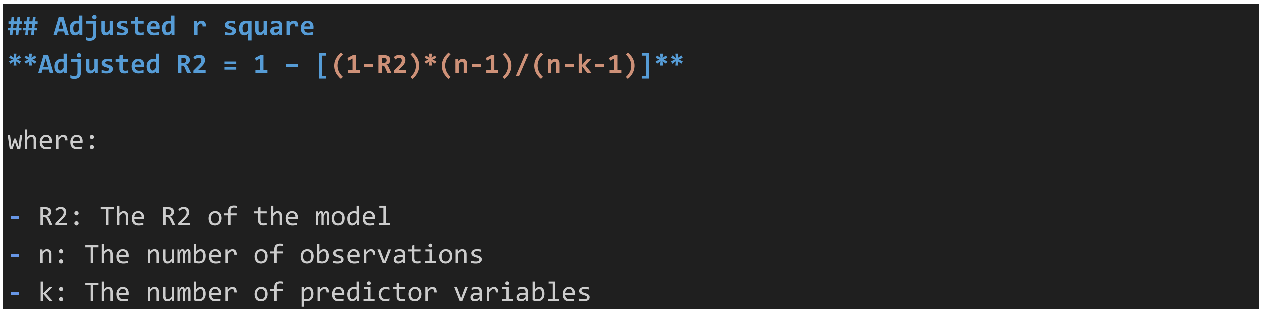
```
## Accuracy of the model R squared and Adjusted r ssquared
## R square
Formula

**R^2 = 1 - SSR/SST**


- R^2   =   Accuracy of the model
- SSR   =   sum of squares of residuals
- SST   =   total sum of squares
```



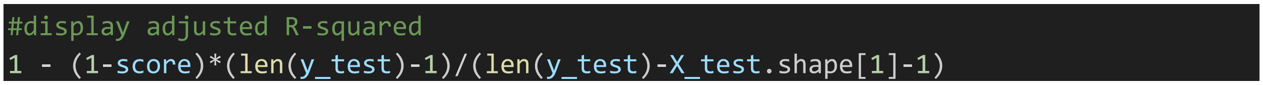
```python
from sklearn.metrics import r2_score
```



```python
score=r2_score(y_test,y_pred_test)
score
```



```
## Adjusted r square
**Adjusted R2 = 1 – [(1-R2)*(n-1)/(n-k-1)]**

where:

- R2: The R2 of the model
- n: The number of observations
- k: The number of predictor variables
```



```python
#display adjusted R-squared
1 - (1-score)*(len(y_test)-1)/(len(y_test)-X_test.shape[1]-1)
```