In [32]:

```python
import pandas as pd
df = pd.read_csv("Advertising.csv")
df
```

Out[32]:

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

200 rows × 4 columns

In [33]:

```python
df.head(5)
```

Out[33]:

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9

In [34]:

```python
df.info
```

Out[34]:

```
<bound method DataFrame.info of         TV  Radio  Newspaper  Sales
0    230.1   37.8       69.2   22.1
1     44.5   39.3       45.1   10.4
2     17.2   45.9       69.3   12.0
3    151.5   41.3       58.5   16.5
4    180.8   10.8       58.4   17.9
..     ...    ...        ...    ...
195   38.2    3.7       13.8    7.6
196   94.2    4.9        8.1   14.0
197  177.0    9.3        6.4   14.8
198  283.6   42.0       66.2   25.5
199  232.1    8.6        8.7   18.4

[200 rows x 4 columns]>
```

In [35]:

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

In [36]:

```
regressor=LinearRegression()
```

In [37]:

```
x=df.iloc[:,:-1]
x
```

Out[37]:

	TV	Radio	Newspaper
0	230.1	37.8	69.2
1	44.5	39.3	45.1
2	17.2	45.9	69.3
3	151.5	41.3	58.5
4	180.8	10.8	58.4
...	...	...	...
195	38.2	3.7	13.8
196	94.2	4.9	8.1
197	177.0	9.3	6.4
198	283.6	42.0	66.2
199	232.1	8.6	8.7

200 rows × 3 columns

In [38]:

```
y=df.iloc[:,-1]
y
```

Out[38]:

```
0      22.1
1      10.4
2      12.0
3      16.5
4      17.9
       ...
195     7.6
196    14.0
197    14.8
198    25.5
199    18.4
Name: Sales, Length: 200, dtype: float64
```

In [39]:

```
x.shape
```

Out[39]:

```
(200, 3)
```

In [40]:

```
y.shape
```

Out[40]:

```
(200,)
```

In [41]:

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.25,random_st
xtrain,xtest,ytrain,ytest
```

Out[41]:

```
(         TV  Radio  Newspaper
 98   289.7   42.3       51.2
 123  123.1   34.6       12.4
 119   19.4   16.0       22.3
 53   182.6   46.2       58.7
 33   265.6   20.0        0.3
 ..     ...    ...        ...
 133  219.8   33.5       45.1
 137  273.7   28.9       59.7
 72    26.8   33.0       19.3
 140   73.4   17.0       12.9
 37    74.7   49.4       45.7

 [150 rows x 3 columns],
          TV  Radio  Newspaper
 58   210.8   49.6       37.7
 40   202.5   22.3       31.6
 34    95.7    1.4        7.4
 102  280.2   10.1       21.4
```

In [42]:

```
xtest.shape
```

Out[42]:

```
(50, 3)
```

In [43]:

```
xtrain.shape
```

Out[43]:

```
(150, 3)
```

In [44]:

```
ytest.shape
```

Out[44]:

```
(50,)
```

In [45]:

```
ytrain.shape
```

Out[45]:

```
(150,)
```

In [48]:

```
regressor.fit(xtrain,ytrain)

```

Out[48]:

```
LinearRegression()
```

In [52]:

```
prediction=regressor.predict(xtest)
results=pd.DataFrame({'Actual':ytest,'Prediction':prediction})
results
```

Out[52]:

	Actual	Prediction
58	23.8	21.291421
40	16.6	18.041942
34	11.9	10.030651
102	19.8	21.048190
184	17.6	20.751671
198	25.5	24.529483
95	16.9	16.859365
4	17.9	15.698007
29	10.5	10.172372
168	17.1	18.902665
171	17.5	15.827508
18	11.3	10.537685
11	17.4	18.862923
89	16.7	15.579691
110	18.4	17.898534
118	15.9	15.359746
159	12.9	13.763299
35	17.8	21.000740
136	9.5	10.030009
59	18.4	19.209740
51	10.7	11.123272
16	12.5	12.181410
44	8.5	8.670379
94	11.5	11.962480
31	11.9	12.633323
162	19.9	16.834847
38	10.1	9.753176
28	18.9	21.064535
193	19.6	18.075131
27	20.9	19.524773
47	23.2	22.044410
165	16.9	17.907452
194	17.3	16.485345
177	16.7	14.791909
176	20.2	21.357310
97	20.5	16.936402
174	16.5	17.187429
73	11.0	12.336883

	Actual	Prediction
69	22.3	21.029747
172	7.6	7.775842
108	5.3	5.413222
107	12.0	9.640061
189	6.7	6.914089
14	19.0	19.224138
56	5.5	7.938005
19	14.6	15.168546
114	14.6	13.731327
39	21.5	21.014193
185	22.6	20.499292
124	19.7	20.577922

In [54]:

```
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_s
```

In [55]:

```
mean_absolute_error(ytest,prediction)
```

Out[55]:

```
1.2187904107011895
```

In [56]:

```
mean_squared_error(ytest,prediction)
```

Out[56]:

```
2.278925158685002
```

In [57]:

```
r2_score(mean_absolute_error,mean_squared_error)
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
<ipython-input-57-33c9f33d0000> in <module>
----> 1 r2_score(mean_absolute_error,mean_squared_error)

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*arg
s, **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

~\anaconda3\lib\site-packages\sklearn\metrics\_regression.py in r2_score(y
_true, y_pred, sample_weight, multioutput)
    674     -3.0
    675     """
--> 676     y_type, y_true, y_pred, multioutput = _check_reg_targets(
    677         y_true, y_pred, multioutput)
    678     check_consistent_length(y_true, y_pred, sample_weight)

~\anaconda3\lib\site-packages\sklearn\metrics\_regression.py in _check_reg
_targets(y_true, y_pred, multioutput, dtype)
     86         the dtype argument passed to check_array.
     87     """
---> 88     check_consistent_length(y_true, y_pred)
     89     y_true = check_array(y_true, ensure_2d=False, dtype=dtype)
     90     y_pred = check_array(y_pred, ensure_2d=False, dtype=dtype)

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in check_consist
ent_length(*arrays)
    257     """
    258 
--> 259     lengths = [_num_samples(X) for X in arrays if X is not None]
    260     uniques = np.unique(lengths)
    261     if len(uniques) > 1:

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in <listcomp>(.
0)
    257     """
    258 
--> 259     lengths = [_num_samples(X) for X in arrays if X is not None]
    260     uniques = np.unique(lengths)
    261     if len(uniques) > 1:

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in _num_samples
(x)
    196             x = np.asarray(x)
    197         else:
--> 198             raise TypeError(message)
    199 
    200     if hasattr(x, 'shape') and x.shape is not None:

TypeError: Expected sequence or array-like, got <class 'function'>
```

In [ ]:

```
```