TASK 5

Sales Prediction

AIM: TO PREDICT HOW MUCH A CUSTOMER CAN BUY BASED ON THE FACTOR SUCH AS AMOUNT SPENT ON THE FOLLOWING FACTORS FOR ADVERTISING:

TV

RADIO

NEWSPAPER

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
D:\Users\Namitha C\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version 1.24.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [2]:
```
data=pd.read_csv("C:/Users/Namitha C/Downloads/Advertising.csv")
data
```

Out[2]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
0	1	230.1	37.8	69.2	22.1
1	2	44.5	39.3	45.1	10.4
2	3	17.2	45.9	69.3	9.3
3	4	151.5	41.3	58.5	18.5
4	5	180.8	10.8	58.4	12.9
...	...	...	...	...	...
195	196	38.2	3.7	13.8	7.6
196	197	94.2	4.9	8.1	9.7
197	198	177.0	9.3	6.4	12.8
198	199	283.6	42.0	66.2	25.5
199	200	232.1	8.6	8.7	13.4

200 rows × 5 columns

DATA EXPLORATION

In [3]:
```
data.head()
```

Out[3]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
0	1	230.1	37.8	69.2	22.1
1	2	44.5	39.3	45.1	10.4
2	3	17.2	45.9	69.3	9.3
3	4	151.5	41.3	58.5	18.5
4	5	180.8	10.8	58.4	12.9

In [4]:
```
data.tail()
```

Out[4]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
195	196	38.2	3.7	13.8	7.6
196	197	94.2	4.9	8.1	9.7
197	198	177.0	9.3	6.4	12.8
198	199	283.6	42.0	66.2	25.5
199	200	232.1	8.6	8.7	13.4

In [5]:
```
data.shape
```

Out[5]:
```
(200, 5)
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Unnamed: 0  200 non-null    int64  
 1   TV          200 non-null    float64
 2   Radio       200 non-null    float64
 3   Newspaper   200 non-null    float64
 4   Sales       200 non-null    float64
dtypes: float64(4), int64(1)
memory usage: 7.9 KB
```

In [7]:
```
data.describe()
```

Out[7]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
count	200.000000	200.000000	200.000000	200.000000	200.000000
mean	100.500000	147.042500	23.264000	30.554000	14.022500
std	57.879185	85.854236	14.846809	21.778621	5.217457
min	1.000000	0.700000	0.000000	0.300000	1.600000
25%	50.750000	74.375000	9.975000	12.750000	10.375000
50%	100.500000	149.750000	22.900000	25.750000	12.900000
75%	150.250000	218.825000	36.525000	45.100000	17.400000
max	200.000000	296.400000	49.600000	114.000000	27.000000

In [8]:
```
data.columns
```

Out[8]:
```
Index(['Unnamed: 0', 'TV', 'Radio', 'Newspaper', 'Sales'], dtype='object')
```

In [9]:
```
data.nunique()
```

Out[9]:
```
Unnamed: 0    200
TV            190
Radio         167
Newspaper     172
Sales         121
dtype: int64
```

CHECK FOR DUPICATES

In [10]:
```
data.duplicated().sum()
```

Out[10]:
```
0
```

CHECK FOR NULL VALUES

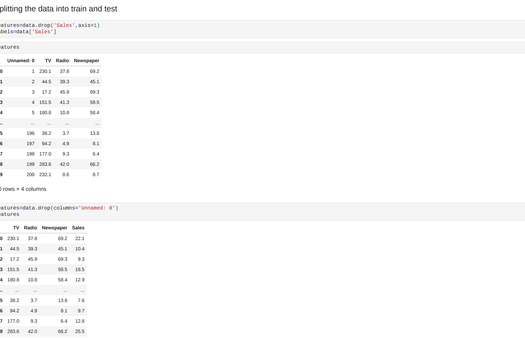
In [11]:
```
data.isnull().sum()
```

Out[11]:
```
Unnamed: 0    0
TV            0
Radio         0
Newspaper     0
Sales         0
dtype: int64
```

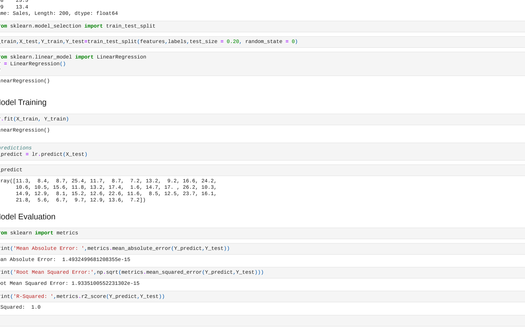
CHECK RELATIONSHIP BETWEEN SALES AND OTHER VARIABLES

In [12]:
```
import plotly.graph_objects as go
```

In [13]:
```
fig=go.Figure()
fig.add_trace(go.Scatter(x=data['TV'],y=data['Sales'],mode='markers',marker=dict(color='blue')))
fig.update_layout(title='TV SALES',xaxis_title='TV', yaxis_title='Sales')
fig.show()
```



In [14]:
```
fig.add_trace(go.Scatter(x=data['Radio'], y=data['Sales'], mode='markers', marker=dict(color='blue')))
fig.update_layout(title='Radio Sales', xaxis_title='Radio', yaxis_title='Sales')
fig.show()
```



In [15]:
```
fig.add_trace(go.Scatter(x=data['Newspaper'], y=data['Sales'], mode='markers', marker=dict(color='blue')))
fig.update_layout(title='Newspaper Sales', xaxis_title='Newspaper', yaxis_title='Sales')
fig.show()
```

Splitting the data into train and test

In [16]:
```
features=data.drop('Sales',axis=1)
labels=data['Sales']
```

In [17]:
```
features
```

Out[17]:

	Unnamed: 0	TV	Radio	Newspaper
0	1	230.1	37.8	69.2
1	2	44.5	39.3	45.1
2	3	17.2	45.9	69.3
3	4	151.5	41.3	58.5
4	5	180.8	10.8	58.4
...	...	...	...	...
195	196	38.2	3.7	13.8
196	197	94.2	4.9	8.1
197	198	177.0	9.3	6.4
198	199	283.6	42.0	66.2
199	200	232.1	8.6	8.7

200 rows × 4 columns

In [18]:
```
features=data.drop(columns='Unnamed: 0')
features
```

Out[18]:

	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	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	9.7
197	177.0	9.3	6.4	12.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	13.4

200 rows × 4 columns

In [21]:
```
labels
```

Out[21]:
```
0      22.1
1      10.4
2       9.3
3      18.5
4      12.9
       ... 
195     7.6
196     9.7
197    12.8
198    25.5
199    13.4
Name: Sales, Length: 200, dtype: float64
```

In [22]:
```
from sklearn.model_selection import train_test_split
```

In [24]:
```
X_train,X_test,Y_train,Y_test=train_test_split(features,labels,test_size = 0.20, random_state = 0)
```

In [25]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr
```

Out[25]:
```
LinearRegression()
```

Model Training

In [26]:
```
lr.fit(X_train, Y_train)
```

Out[26]:
```
LinearRegression()
```

In [27]:
```
#predictions
Y_predict = lr.predict(X_test)
```

In [30]:
```
Y_predict
```

Out[30]:
```
array([11.3,  8.4,  8.7, 25.4, 11.7,  8.7,  7.2, 13.2,  9.2, 16.6, 24.2,
       10.6, 10.5, 15.6, 11.8, 13.2, 17.4,  1.6, 14.7, 17. , 20.2, 10.3,
       14.9, 12.9,  8.1, 15.2, 12.9, 22.6, 11.6,  8.9, 12.5, 23.7, 16.1,
       21.8,  5.6,  6.7,  9.7, 12.9, 13.6,  7.2])
```

Model Evaluation

In [31]:
```
from sklearn import metrics
```

In [32]:
```
print('Mean Absolute Error: ',metrics.mean_absolute_error(Y_predict,Y_test))
```

```
Mean Absolute Error:  1.4932499681208355e-15
```

In [33]:
```
print('Root Mean Squared Error:',np.sqrt(metrics.mean_squared_error(Y_predict,Y_test)))
```

```
Root Mean Squared Error: 1.9335106552313192e-15
```

In [35]:
```
print('R-Squared: ',metrics.r2_score(Y_predict,Y_test))
```

```
R-Squared:  1.0
```

In [ ]:
```

```