In [47]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [81]:

```
cd = pd.read_csv('Company_Data (1).csv')
cd
```

Out[81]:

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	Age	Education	U
0	9.50	138	73	11	276	120	Bad	42	17	
1	11.22	111	48	16	260	83	Good	65	10	
2	10.06	113	35	10	269	80	Medium	59	12	
3	7.40	117	100	4	466	97	Medium	55	14	
4	4.15	141	64	3	340	128	Bad	38	13	
...	...	...	...	...	...	...	...	...	...	
395	12.57	138	108	17	203	128	Good	33	14	
396	6.14	139	23	3	37	120	Medium	55	11	
397	7.41	162	26	12	368	159	Medium	40	18	
398	5.94	100	79	7	284	95	Bad	50	12	
399	9.71	134	37	0	27	120	Good	49	16	

400 rows × 11 columns

In [82]:

```
# Initial analysis
```

In [83]:

```
cd.shape
```

Out[83]:

```
(400, 11)
```

In [84]:

```
cd.dtypes
```

Out[84]:

```
Sales          float64
CompPrice        int64
Income           int64
Advertising      int64
Population       int64
Price            int64
ShelveLoc       object
Age              int64
Education        int64
Urban           object
US              object
dtype: object
```

In [72]:

```
cd.isna().sum()
```

Out[72]:

```
Sales          0
CompPrice      0
Income         0
Advertising    0
Population     0
Price          0
ShelveLoc      0
Age            0
Education      0
Urban          0
US             0
dtype: int64
```

In [85]:

```
cd.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Sales        400 non-null    float64
 1   CompPrice    400 non-null    int64  
 2   Income       400 non-null    int64  
 3   Advertising  400 non-null    int64  
 4   Population   400 non-null    int64  
 5   Price        400 non-null    int64  
 6   ShelveLoc    400 non-null    object 
 7   Age          400 non-null    int64  
 8   Education    400 non-null    int64  
 9   Urban        400 non-null    object 
 10  US           400 non-null    object 
dtypes: float64(1), int64(7), object(3)
memory usage: 34.5+ KB
```

In [86]:

```
cd.describe()
```

Out[86]:

	Sales	CompPrice	Income	Advertising	Population	Price	Age	E
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	40
mean	7.496325	124.975000	68.657500	6.635000	264.840000	115.795000	53.322500	1
std	2.824115	15.334512	27.986037	6.650364	147.376436	23.676664	16.200297	
min	0.000000	77.000000	21.000000	0.000000	10.000000	24.000000	25.000000	1
25%	5.390000	115.000000	42.750000	0.000000	139.000000	100.000000	39.750000	1
50%	7.490000	125.000000	69.000000	5.000000	272.000000	117.000000	54.500000	1
75%	9.320000	135.000000	91.000000	12.000000	398.500000	131.000000	66.000000	1
max	16.270000	175.000000	120.000000	29.000000	509.000000	191.000000	80.000000	1

In [89]:

```
cd.dtypes
```

Out[89]:

```
Sales          float64
CompPrice        int64
Income           int64
Advertising      int64
Population       int64
Price            int64
ShelveLoc       object
Age              int64
Education        int64
Urban           object
US              object
dtype: object
```

In [90]:

```
# converting target variable 'Sales' into categories Low, Medium and High.
cd['Sales'] = pd.cut(x=cd['Sales'], bins = [0,6,12,17], labels=['Low','Medium','High'], rig
cd['Sales']
```

Out[90]:

```
0      Medium
1      Medium
2      Medium
3      Medium
4         Low
        ...
395      High
396    Medium
397    Medium
398       Low
399    Medium
Name: Sales, Length: 400, dtype: category
Categories (3, object): ['Low' < 'Medium' < 'High']
```

In [91]:

```
cd['Sales'].value_counts()
```

Out[91]:

```
Medium    243
Low       130
High       27
Name: Sales, dtype: int64
```

In [92]:

```
# converting other attributes into categories
```

In [93]:

```
cd['CompPrice']=pd.cut(x=cd['CompPrice'], bins=[77,100,133,176], labels=['Low','Medium','Hi
cd['Income']=pd.cut(x=cd['Income'], bins=[21,46,71,121],labels=['Low','Medium','High'], rig
cd['Advertising']=pd.cut(x=cd['Advertising'], bins=[0,10,20,30], labels=['Low','Medium','Hi
cd['Population']=pd.cut(x=cd['Population'], bins=[10,170,340,510],labels=['Low','Medium','H
cd['Price']=pd.cut(x=cd['Price'],bins=[24,80,136,192],labels=['Low','Medium','High'],right=
cd['Age']=pd.cut(x=cd['Age'], bins=[25,45,60,81], labels=['Low','Medium','High'],right=Fals
cd['Education']=pd.cut(x=cd['Education'], bins=[10,12.5,15,19], labels=['Low','Medium','Hig
```

In [94]:

```
cd.head()
```

Out[94]:

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	Age	Educatio
0	Medium	High	High	Medium	Medium	Medium	Bad	Low	Hig
1	Medium	Medium	Medium	Medium	Medium	Medium	Good	High	Lo
2	Medium	Medium	Low	Medium	Medium	Medium	Medium	Medium	Lo
3	Medium	Medium	High	Low	High	Medium	Medium	Medium	Mediu
4	Low	High	Medium	Low	High	Medium	Bad	Low	Mediu

In [95]:

```
cd.isna().sum()
```

Out[95]:

```
Sales          0
CompPrice      0
Income         0
Advertising    0
Population     0
Price          0
ShelveLoc      0
Age            0
Education      0
Urban          0
US             0
dtype: int64
```

In [18]:

```
# Encoding Categorial data
```

In [23]:

```
from sklearn.preprocessing import LabelEncoder
```

In [97]:

```
le = LabelEncoder()
```

In [98]:

```
cd['Sales']=le.fit_transform(cd['Sales'])
cd['CompPrice']=le.fit_transform(cd['CompPrice'])
cd['Income']=le.fit_transform(cd['Income'])
cd['Advertising']=le.fit_transform(cd['Advertising'])
cd['Population']=le.fit_transform(cd['Population'])
cd['Price']=le.fit_transform(cd['Price'])
cd['ShelveLoc']=le.fit_transform(cd['ShelveLoc'])
cd['Age']=le.fit_transform(cd['Age'])
cd['Education']= le.fit_transform(cd['Education'])
cd['Urban']= le.fit_transform(cd['Urban'])
cd['US'] = le.fit_transform(cd['US'])
cd
```

Out[98]:

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	Age	Education	U
0	2	0	0	2	2	2	0	1	0	
1	2	2	2	2	2	2	1	0	1	
2	2	2	1	2	2	2	2	2	1	
3	2	2	0	1	0	2	2	2	2	
4	1	0	2	1	0	2	0	1	2	
...	...	...	...	...	...	...	...	...	...	
395	0	0	0	2	2	2	1	1	2	
396	2	0	1	1	1	2	2	2	1	
397	2	0	1	2	0	0	2	1	0	
398	1	2	0	1	2	2	0	2	1	
399	2	0	1	1	1	2	1	2	0	

400 rows × 11 columns

In [99]:

```
# Dividing Data into independent variables and dependent variable
X= cd.drop('Sales', axis=1)
y=cd['Sales']
```

In [103]:

```
# splitting data into training and testing data
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=12)
```

In [104]:

```
X_train.shape,y_train.shape
```

Out[104]:

```
((320, 10), (320,))
```

In [105]:

```
X_test.shape,y_test.shape
```

Out[105]:

```
((80, 10), (80,))
```

In [108]:

```
from sklearn.ensemble import RandomForestClassifier
rf_model = RandomForestClassifier(n_estimators=100, max_depth=3)
rf_model.fit(X_train,y_train)
```

Out[108]:

```
RandomForestClassifier(max_depth=3)
```

In [109]:

```
y_test_pred=rf_model.predict(X_test)
y_test_pred
```

Out[109]:

```
array([1, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2, 1, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 1, 2, 2, 2, 2, 1, 1, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2])
```

In [121]:

```
rf_model.score(X_test,y_test)
```

Out[121]:

```
0.7125
```

In [118]:

```
from sklearn.metrics import accuracy_score, confusion_matrix
```

In [119]:

```
accuracy_score(y_test,y_test_pred)
```

Out[119]:

```
0.7125
```

In [120]:

```
confusion_matrix(y_test,y_test_pred)
```

Out[120]:

```
array([[ 0,  0,  3],
       [ 0,  9, 17],
       [ 0,  3, 48]], dtype=int64)
```

In [ ]:

```

```