Import Libraries

In [1]:

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
```

In [3]:

```
company_data = pd.read_csv('Company_Data.csv')
company_data
```

Out[3]:

	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 [4]:

```
company_data.shape
```

Out[4]:

```
(400, 11)
```

In [5]:

```
company_data.head()
```

Out[5]:

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	Age	Education	Urba
0	9.50	138	73	11	276	120	Bad	42	17	Y
1	11.22	111	48	16	260	83	Good	65	10	Y
2	10.06	113	35	10	269	80	Medium	59	12	Y
3	7.40	117	100	4	466	97	Medium	55	14	Y
4	4.15	141	64	3	340	128	Bad	38	13	Y

In [6]:

```
company_data.dtypes
```

Out[6]:

```
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 [7]:

```
company_data.isna().sum()
```

Out[7]:

```
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 [11]:

```
company_data.describe(include='all')
```

Out[11]:

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400	4
unique	NaN	NaN	NaN	NaN	NaN	NaN	3	
top	NaN	NaN	NaN	NaN	NaN	NaN	Medium	
freq	NaN	NaN	NaN	NaN	NaN	NaN	219	
mean	7.496325	124.975000	68.657500	6.635000	264.840000	115.795000	NaN	
std	2.824115	15.334512	27.986037	6.650364	147.376436	23.676664	NaN	
min	0.000000	77.000000	21.000000	0.000000	10.000000	24.000000	NaN	
25%	5.390000	115.000000	42.750000	0.000000	139.000000	100.000000	NaN	
50%	7.490000	125.000000	69.000000	5.000000	272.000000	117.000000	NaN	
75%	9.320000	135.000000	91.000000	12.000000	398.500000	131.000000	NaN	
max	16.270000	175.000000	120.000000	29.000000	509.000000	191.000000	NaN	

Converting form Categorical data

In [20]:

```
company_data['High']=company_data.Sales.map(lambda x:1if x>8 else 0)
company_data['ShelveLoc']=company_data['ShelveLoc'].astype('category')
company_data['Urban']=company_data['Urban'].astype('category')
company_data['US']=company_data['US'].astype('category')
company_data.dtypes
```

Out[20]:

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

In [21]:

```
company_data.head()
```

Out[21]:

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	Age	Education	Urba
0	9.50	138	73	11	276	120	Bad	42	17	Y
1	11.22	111	48	16	260	83	Good	65	10	Y
2	10.06	113	35	10	269	80	Medium	59	12	Y
3	7.40	117	100	4	466	97	Medium	55	14	Y
4	4.15	141	64	3	340	128	Bad	38	13	Y

In [22]:

```
# Label Encoding
company_data['ShelveLoc']=company_data['ShelveLoc'].cat.codes
company_data['Urban']=company_data['Urban'].cat.codes
company_data['US']=company_data['US'].cat.codes
company_data.tail()
```

Out[22]:

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	Age	Education	U
395	12.57	138	108	17	203	128	1	33	14	
396	6.14	139	23	3	37	120	2	55	11	
397	7.41	162	26	12	368	159	2	40	18	
398	5.94	100	79	7	284	95	0	50	12	
399	9.71	134	37	0	27	120	1	49	16	

Setting feature and target variable

In [23]:

```
feature_colm=['Income','CompPrice','Advertising','Population','Price','ShelveLoc','Age','Ed
x=company_data[feature_colm]
y=company_data.High
print(x)
print(y)
```

```
     Income  CompPrice  Advertising  Population  Price  ShelveLoc  Age  \
0        73        138           11         276    120          0   42
1        48        111           16         260     83          1   65
2        35        113           10         269     80          2   59
3       100        117            4         466     97          2   55
4        64        141            3         340    128          0   38
..      ...        ...          ...         ...    ...        ...  ...
395     108        138           17         203    128          1   33
396      23        139            3          37    120          2   55
397      26        162           12         368    159          2   40
398      79        100            7         284     95          0   50
399      37        134            0          27    120          1   49

     Education  Urban  US
0           17      1   1
1           10      1   1
2           12      1   1
3           14      1   1
4           13      1   0
..         ...    ...  ..
395         14      1   1
396         11      0   1
397         18      1   1
398         12      1   1
399         16      1   1

[400 rows x 10 columns]
0      1
1      1
2      1
3      0
4      0
      ..
395    1
396    0
397    0
398    0
399    1
Name: High, Length: 400, dtype: int64
```

In [24]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
x_train,x_test,y_train,y_test
```

Out[24]:

```
(     Income  CompPrice  Advertising  Population  Price  ShelveLoc  Age  \
 336      35        138            6          60    143          0   28
 64       67        100           12         184    104          2   32
 55       81        143            5          60    154          2   61
 106      33        102            0         217    139          2   70
 300      78        116            1         158     99          2   45
 ..      ...        ...          ...         ...    ...        ...  ...
 323     105        107           18         428    103          2   34
 192      26        108            0         408     93          2   56
 117      53        145            0         507    119          2   41
 47       98        126            0         173    108          0   55
 172     102        104           13         123    110          1   35

      Education  Urban  US
 336         18      1   0
 64          16      0   1
 55          18      1   1
 106         18      0   0
 300         11      1   1
 ..         ...    ...  ..
 323         12      1   1
 192         14      0   0
 117         12      1   0
 47          16      1   0
 172         16      1   1

 [320 rows x 10 columns],
      Income  CompPrice  Advertising  Population  Price  ShelveLoc  Age  \
 132      87        125            9         232    136          1   72
 309     111        131           13          33     80          0   68
 341     120         98            0         268     93          2   72
 196      28        130            6         410    133          0   72
 246      56        120           20         266     90          0   78
 ..      ...        ...          ...         ...    ...        ...  ...
 14      117        107           11         148    118          1   52
 363      75        111            1         377    108          1   25
 304      98        123           12         408    134          1   29
 361      25        131           10         183    104          2   56
 329      54        100            9         433     89          1   45

      Education  Urban  US
 132         10      1   1
 309         18      1   1
 341         10      0   0
 196         16      1   1
 246         18      1   1
 ..         ...    ...  ..
 14          18      1   1
 363         12      1   0
 304         10      1   1
 361         15      0   1
 329         12      1   1

 [80 rows x 10 columns],
```

```
336    0
64     0
55     0
106    0
300    1
      ..
323    1
192    0
117    1
47     0
172    1
Name: High, Length: 320, dtype: int64,
132    1
309    1
341    0
196    0
246    0
      ..
14     1
363    1
304    1
361    1
329    1
Name: High, Length: 80, dtype: int64)
```

Building Decision Tree

In [25]:

```
dc_model=BaggingClassifier(DecisionTreeClassifier(max_depth=8),random_state=0)
dc_model=AdaBoostClassifier(DecisionTreeClassifier(max_depth=8),random_state=0)
dc_model=dc_model.fit(x_train,y_train)
y_pred=dc_model.predict(x_test)
```

In [27]:

```
# Accuracy
accuracy_score(y_test,y_pred)
```

Out[27]:

```
0.6875
```

In [30]:

```
print(confusion_matrix(y_test,y_pred))
```

```
[[36  7]
 [18 19]]
```

In [31]:

```
print(classification_report(y_test,y_pred))
```

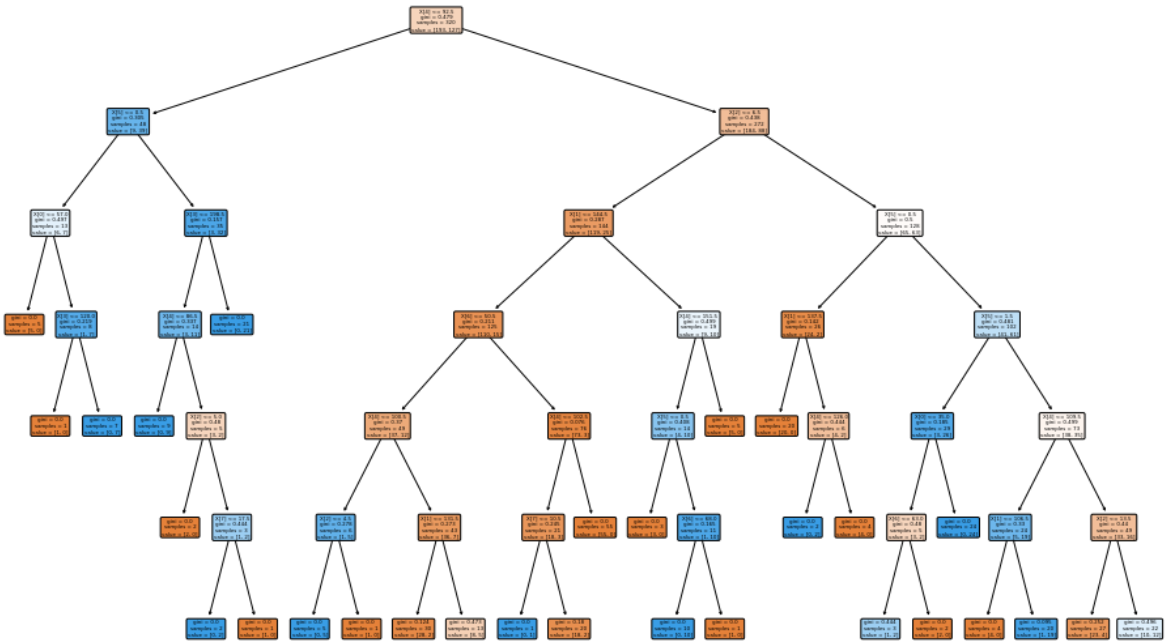
```
              precision    recall  f1-score   support

           0       0.67      0.84      0.74        43
           1       0.73      0.51      0.60        37

    accuracy                           0.69        80
   macro avg       0.70      0.68      0.67        80
weighted avg       0.70      0.69      0.68        80
```

In [32]:

```
dc_model=DecisionTreeClassifier(criterion='gini',max_depth=6)
dc_model=dc_model.fit(x_train,y_train)
```

In [33]:

```
from sklearn.tree import plot_tree
from matplotlib import pyplot as plt
plt.figure(figsize=(16,10))
plot_tree(dc_model,rounded=True,filled=True)
plt.show()
```



In [ ]:

```

```