In [42]:
```python
import numpy as ny
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [43]:
```python
df=pd.read_csv(r"C:\Users\rubin\Downloads\drug200.csv")
df
```

Out[43]:

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	HIGH	HIGH	25.355	drugY
1	47	M	LOW	HIGH	13.093	drugC
2	47	M	LOW	HIGH	10.114	drugC
3	28	F	NORMAL	HIGH	7.798	drugX
4	61	F	LOW	HIGH	18.043	drugY
...	...	...	...	...	...	...
195	56	F	LOW	HIGH	11.567	drugC
196	16	M	LOW	HIGH	12.006	drugC
197	52	M	NORMAL	HIGH	9.894	drugX
198	23	M	NORMAL	NORMAL	14.020	drugX
199	40	F	LOW	NORMAL	11.349	drugX

200 rows × 6 columns

In [44]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Age          200 non-null    int64
 1   Sex          200 non-null    object
 2   BP           200 non-null    object
 3   Cholesterol  200 non-null    object
 4   Na_to_K      200 non-null    float64
 5   Drug         200 non-null    object
dtypes: float64(1), int64(1), object(4)
memory usage: 9.5+ KB
```

In [45]:
```python
df['Age'].value_counts()
```

Out[45]:   Age
            47   8
            23   7
            28   7
            49   7
            39   6
            32   6
            50   5
            37   5
            58   5
            60   5
            22   5
            34   4
            72   4
            51   4
            42   4
            26   4
            24   4
            74   4
            67   4
            68   4
            61   4
            56   4
            20   4
            36   4
            45   4
            41   4
            31   4
            43   4
            65   4
            57   4
            53   3
            40   3
            70   3
            59   3
            16   3
            38   3
            15   3
            69   3
            35   3
            18   3
            64   3
            52   2
            55   2
            62   2
            19   2
            29   2
            66   2
            73   2
            46   2
            48   2
            54   1
            17   1
            33   1
            63   1
            30   1
            21   1
            25   1
            Name: count, dtype: int64

In [46]:
```python
df['BP'].value_counts()
```

Out[46]:
```
BP
HIGH      77
LOW       64
NORMAL    59
Name: count, dtype: int64
```

In [47]:
```python
df['Cholesterol'].value_counts()
```

Out[47]:
```
Cholesterol
HIGH      103
NORMAL     97
Name: count, dtype: int64
```

In [48]:
```python
convert={"BP":{"HIGH":1,"NORMAL":2,"LOW":3}}
df=df.replace(convert)
df
```

Out[48]:

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	1	HIGH	25.355	drugY
1	47	M	3	HIGH	13.093	drugC
2	47	M	3	HIGH	10.114	drugC
3	28	F	2	HIGH	7.798	drugX
4	61	F	3	HIGH	18.043	drugY
...	...	...	...	...	...	...
195	56	F	3	HIGH	11.567	drugC
196	16	M	3	HIGH	12.006	drugC
197	52	M	2	HIGH	9.894	drugX
198	23	M	2	NORMAL	14.020	drugX
199	40	F	3	NORMAL	11.349	drugX

200 rows × 6 columns

In [60]:
```python
convert={"Drug":{"drugX":1,"drugY":2,"drugA":3,"drugB":4,"drugC":5}}
df=df.replace(convert)
df
```

Out[60]:

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	1	HIGH	25.355	2
1	47	M	3	HIGH	13.093	3
2	47	M	3	HIGH	10.114	3
3	28	F	2	HIGH	7.798	1
4	61	F	3	HIGH	18.043	2
...	...	...	...	...	...	...
195	56	F	3	HIGH	11.567	3
196	16	M	3	HIGH	12.006	3
197	52	M	2	HIGH	9.894	1
198	23	M	2	NORMAL	14.020	1
199	40	F	3	NORMAL	11.349	1

200 rows × 6 columns

In [61]:
```python
x=["Drug","BP"]
y=["M","F"]
all_inputs=df[x]
all_classes=df["Sex"]
```

In [62]:
```python
(x_train,x_test,y_train,y_test)=train_test_split(all_inputs,all_classes,test_size=0.2
```

In [63]:
```python
clf=DecisionTreeClassifier(random_state=0)
```

In [64]:
```python
clf.fit(x_train,y_train)
```

Out[64]:
```
▼          DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

In [65]:
```python
score=clf.score(x_test,y_test)
print(score)
```

0.46

In [ ]:

In [ ]: