

```
In [1]: import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

```
In [2]: df=pd.read_csv(r"C:\Users\LENOVO\Downloads\drug200.csv")
df
```

Out[2]:

	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 [3]: df.shape
```

Out[3]: (200, 6)

```
In [4]: df.head()
```

Out[4]:

	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

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

Out[5]: (200, 6)

In [6]: `df.describe`

```
Out[6]: <bound method NDFrame.describe of
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 x 6 columns]>
```

In [7]: `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 [8]: `df.isna().sum()`

```
Out[8]: Age             0
Sex             0
BP              0
Cholesterol     0
Na_to_K         0
Drug            0
dtype: int64
```

In [9]: `df['BP'].value_counts()`

```
Out[9]: BP
HIGH      77
LOW       64
NORMAL    59
Name: count, dtype: int64
```

In [10]: `df['Cholesterol'].value_counts()`

```
Out[10]: Cholesterol
HIGH      103
NORMAL    97
Name: count, dtype: int64
```

```
In [11]: df['Drug'].value_counts()
```

```
Out[11]: Drug
drugY    91
drugX    54
drugA    23
drugC    16
drugB    16
Name: count, dtype: int64
```

```
In [12]: df['Na_to_K'].value_counts()
```

```
Out[12]: Na_to_K
12.006    2
18.295    2
25.355    1
11.939    1
16.347    1
..
24.658    1
24.276    1
13.967    1
19.675    1
11.349    1
Name: count, Length: 198, dtype: int64
```

```
In [13]: convert={"Sex":{"F":1,"M":0}}
df=df.replace(convert)
df
```

```
Out[13]:
```

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

200 rows × 6 columns

```
In [14]: convert={"Cholesterol":{"HIGH":1,"NORMAL":0}}
df=df.replace(convert)
df
```

Out[14]:

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

200 rows × 6 columns

```
In [15]: x=["Sex","Cholesterol"]
y=["Yes","No"]
all_inputs=df[x]
all_classes=df["BP"]
```

```
In [16]: (x_train,x_test,y_train,y_test)=train_test_split(all_inputs,all_classes,test_size=0.25)
```

```
In [17]: clf=DecisionTreeClassifier(random_state=0)
```

```
In [18]: clf.fit(x_train,y_train)
```

Out[18]:

▼

DecisionTreeClassifier
DecisionTreeClassifier(random\_state=0)

```
In [20]: score=clf.score(x_test,y_test)
print(score)
```

0.38

In [ ]: