In [76]:
```
import pandas as pd
import numpy as np
df = pd.read_csv("car_evaluation.csv")
df
```

Out[76]:

	buying	maint	doors	persons	lug_boot	safety	outcome
0	vhigh	vhigh	2	2	small	low	unacc
1	vhigh	vhigh	2	2	small	med	unacc
2	vhigh	vhigh	2	2	small	high	unacc
3	vhigh	vhigh	2	2	med	low	unacc
4	vhigh	vhigh	2	2	med	med	unacc
...	...	...	...	...	...	...	...
1723	low	low	5	5	med	med	good
1724	low	low	5	5	med	high	vgood
1725	low	low	5	5	big	low	unacc
1726	low	low	5	5	big	med	good
1727	low	low	5	5	big	high	vgood

1728 rows × 7 columns

In [77]:
```
x = df.iloc[:,:-1]
y = df.iloc[:,-1]
x
```

Out[77]:

	buying	maint	doors	persons	lug_boot	safety
0	vhigh	vhigh	2	2	small	low
1	vhigh	vhigh	2	2	small	med
2	vhigh	vhigh	2	2	small	high
3	vhigh	vhigh	2	2	med	low
4	vhigh	vhigh	2	2	med	med
...	...	...	...	...	...	...
1723	low	low	5	5	med	med
1724	low	low	5	5	med	high
1725	low	low	5	5	big	low
1726	low	low	5	5	big	med
1727	low	low	5	5	big	high

1728 rows × 6 columns

In [78]:
```
df.outcome.value_counts()
```

Out[78]:
```
unacc    1210
acc       384
good       69
vgood      65
Name: outcome, dtype: int64
```

In [79]:
```
from sklearn.preprocessing import LabelEncoder
label = LabelEncoder()
```

In [80]:
```
x[["buying","maint","lug_boot","safety"]]= x[["buying","maint","lug_boot","safety"]].apply(label.fit_transform)

# x['maint'].unique()
```

In [81]:
```
x
```

Out[81]:

	buying	maint	doors	persons	lug_boot	safety
0	3	3	2	2	2	1
1	3	3	2	2	2	2
2	3	3	2	2	2	0
3	3	3	2	2	1	1
4	3	3	2	2	1	2
...	...	...	...	...	...	...
1723	1	1	5	5	1	2
1724	1	1	5	5	1	0
1725	1	1	5	5	0	1
1726	1	1	5	5	0	2
1727	1	1	5	5	0	0

1728 rows × 6 columns

In [82]:
```
from sklearn.model_selection import train_test_split
x_train, x_test , y_train, y_test = train_test_split(x,y,test_size = 0.25)
```

In [83]:
```
print(x_train.shape,x_test.shape)
```
```
(1296, 6) (432, 6)
```

In [84]:
```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier()
```

In [85]:
```
model.fit(x_train,y_train)
```

Out[85]:
```
KNeighborsClassifier()
```

In [86]:
```
y_pred = model.predict(x_test)
```

In [87]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)*100
```

Out[87]:
```
91.20370370370371
```

In [88]:
```
pd.crosstab(y_test,y_pred)
```

Out[88]:

col_0	acc	good	unacc	vgood
outcome				
acc	88	2	18	0
good	10	4	0	2
unacc	2	0	292	0
vgood	4	0	0	10

In [89]:
```
from imblearn.over_sampling import SMOTE
smote = SMOTE()
```

In [90]:
```
x_train_smoted,y_train_smoted = smote.fit_resample(x_train.astype("float"),y_train)
```

In [91]:
```
from collections import Counter
print("Before Smoting",Counter(y_train))
print("Before Smoting",Counter(y_train_smoted))
```
```
Before Smoting Counter({'unacc': 916, 'acc': 276, 'good': 53, 'vgood': 51})
Before Smoting Counter({'unacc': 916, 'good': 916, 'acc': 916, 'vgood': 916})
```

In [92]:
```
model.fit(x_train_smoted,y_train_smoted)
```

Out[92]:
```
KNeighborsClassifier()
```

In [93]:
```
y_pred_smoted = model.predict(x_test)
```

In [102…
```
accuracy_score(y_test,y_pred_smoted)*100
```

Out[102…
```
93.28703703703704
```

In [103…
```
pd.crosstab(y_test,y_pred_smoted)
```

Out[103…

col_0	acc	good	unacc	vgood
outcome				
acc	91	6	11	0
good	1	14	0	1
unacc	7	2	285	0
vgood	1	0	0	13

In [ ]:
```

```