In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree  import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
wine=pd.read_csv('wine.csv')
```

In [3]:

```
wine.head()
```

Out[3]:

	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD280/OD: of dilu wi
0	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3
1	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3
2	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3
3	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3
4	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2

In [4]:

```
wine.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 178 entries, 0 to 177
Data columns (total 14 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Class                         178 non-null    int64  
 1   Alcohol                       178 non-null    float64
 2   Malic acid                    178 non-null    float64
 3   Ash                           178 non-null    float64
 4   Alcalinity of ash             178 non-null    float64
 5   Magnesium                     178 non-null    int64  
 6   Total phenols                 178 non-null    float64
 7   Flavanoids                    178 non-null    float64
 8   Nonflavanoid phenols          178 non-null    float64
 9   Proanthocyanins               178 non-null    float64
 10  Color intensity               178 non-null    float64
 11  Hue                           178 non-null    float64
 12  OD280/OD315 of diluted wines  178 non-null    float64
 13  Proline                       178 non-null    int64  
dtypes: float64(11), int64(3)
memory usage: 19.6 KB
```

In [5]:

```
wine.shape
```

Out[5]:

```
(178, 14)
```

In [6]:

```
wine.dtypes
```

Out[6]:

```
Class                             int64
Alcohol                         float64
Malic acid                      float64
Ash                             float64
Alcalinity of ash               float64
Magnesium                         int64
Total phenols                   float64
Flavanoids                      float64
Nonflavanoid phenols            float64
Proanthocyanins                 float64
Color intensity                 float64
Hue                             float64
OD280/OD315 of diluted wines    float64
Proline                           int64
dtype: object
```

In [7]:

```
wine.columns
```

Out[7]:

```
Index(['Class', 'Alcohol', 'Malic acid', 'Ash', 'Alcalinity of ash',
       'Magnesium', 'Total phenols', 'Flavanoids', 'Nonflavanoid phenols',
       'Proanthocyanins', 'Color intensity', 'Hue',
       'OD280/OD315 of diluted wines', 'Proline'],
      dtype='object')
```

In [8]:

```
wine.describe()
```

Out[8]:

	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanin
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.00000
mean	1.938202	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.029270	0.361854	1.59089
std	0.775035	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.998859	0.124453	0.57235
min	1.000000	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.340000	0.130000	0.41000
25%	1.000000	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.205000	0.270000	1.25000
50%	2.000000	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.135000	0.340000	1.55500
75%	3.000000	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.875000	0.437500	1.95000
max	3.000000	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.080000	0.660000	3.58000

In [9]:

```
#to understand it has two or more unique variables and is it regression or classification
wine.Class.unique()
```

Out[9]:

```
array([1, 2, 3], dtype=int64)
```

In [10]:

```
# identifing null values
wine.isnull().sum()
```

Out[10]:

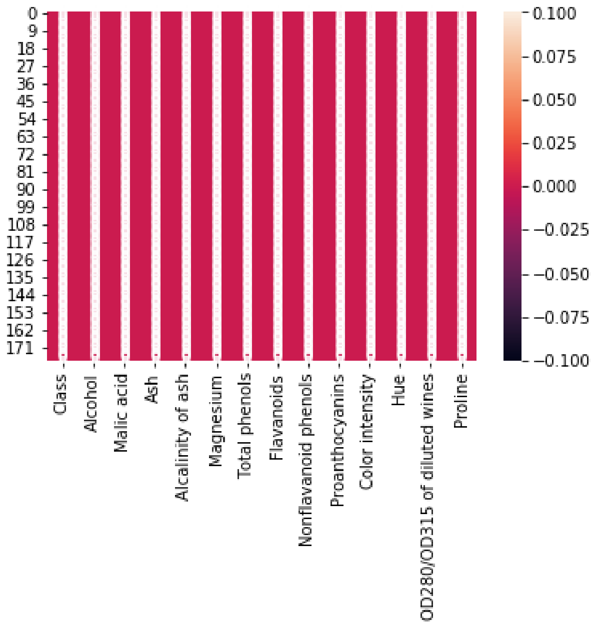
```
Class                           0
Alcohol                         0
Malic acid                      0
Ash                             0
Alcalinity of ash               0
Magnesium                       0
Total phenols                   0
Flavanoids                      0
Nonflavanoid phenols            0
Proanthocyanins                 0
Color intensity                 0
Hue                             0
OD280/OD315 of diluted wines    0
Proline                         0
dtype: int64
```

In [11]:

```
sns.heatmap(wine.isnull(),annot=True)
```

Out[11]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22f607450c8>
```



In [12]:

```
wine.corr()
```

Out[12]:

	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins
Class	1.000000	-0.328222	0.437776	-0.049643	0.517859	-0.209179	-0.719163	-0.847498	0.489109	-0.499130
Alcohol	-0.328222	1.000000	0.094397	0.211545	-0.310235	0.270798	0.289101	0.236815	-0.155929	0.136698
Malic acid	0.437776	0.094397	1.000000	0.164045	0.288500	-0.054575	-0.335167	-0.411007	0.292977	-0.220746
Ash	-0.049643	0.211545	0.164045	1.000000	0.443367	0.286587	0.128980	0.115077	0.186230	0.009652

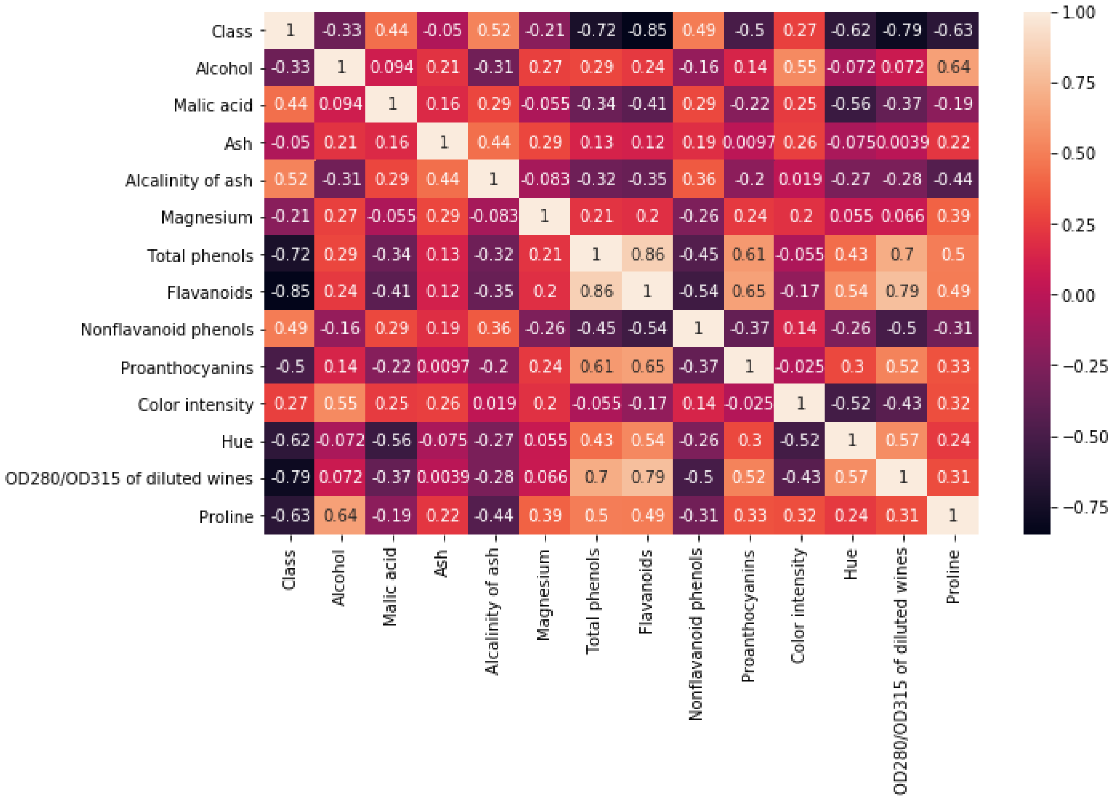
	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins
Alcalinity of ash	0.517859	-0.310235	0.288500	0.443367	1.000000	-0.083333	-0.321113	-0.351370	0.361922	-0.197327
Magnesium	-0.209179	0.270798	-0.054575	0.286587	-0.083333	1.000000	0.214401	0.195784	-0.256294	0.236441
Total phenols	-0.719163	0.289101	-0.335167	0.128980	-0.321113	0.214401	1.000000	0.864564	-0.449935	0.612413
Flavanoids	-0.847498	0.236815	-0.411007	0.115077	-0.351370	0.195784	0.864564	1.000000	-0.537900	0.652692
Nonflavanoid phenols	0.489109	-0.155929	0.292977	0.186230	0.361922	-0.256294	-0.449935	-0.537900	1.000000	-0.365845
Proanthocyanins	-0.499130	0.136698	-0.220746	0.009652	-0.197327	0.236441	0.612413	0.652692	-0.365845	1.000000
Color intensity	0.265668	0.546364	0.248985	0.258887	0.018732	0.199950	-0.055136	-0.172379	0.139057	-0.025250
Hue	-0.617369	-0.071747	-0.561296	-0.074667	-0.273955	0.055398	0.433681	0.543479	-0.262640	0.295544
OD280/OD315 of diluted wines	-0.788230	0.072343	-0.368710	0.003911	-0.276769	0.066004	0.699949	0.787194	-0.503270	0.519067
Proline	-0.633717	0.643720	-0.192011	0.223626	-0.440597	0.393351	0.498115	0.494193	-0.311385	0.330417

In [13]:

```
plt.figure(figsize=(10,6))
sns.heatmap(wine.corr(),annot=True)
```

Out[13]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22f623b8b48>
```



In [14]:

```
wine.skew()
```

Out[14]:

```
Class                      0.107431
Alcohol                   -0.051482
Malic acid                 1.039651
```

```
Ash                            -0.176699
Alcalinity of ash               0.213047
Magnesium                       1.098191
Total phenols                   0.086639
Flavanoids                      0.025344
Nonflavanoid phenols            0.450151
Proanthocyanins                 0.517137
Color intensity                 0.868585
Hue                             0.021091
OD280/OD315 of diluted wines   -0.307285
Proline                         0.767822
dtype: float64
```

In [15]:

```python
for col in wine.columns:
    if wine.skew().loc[col]>0.55:
        wine[col]=np.log1p(wine[col])
```

In [16]:

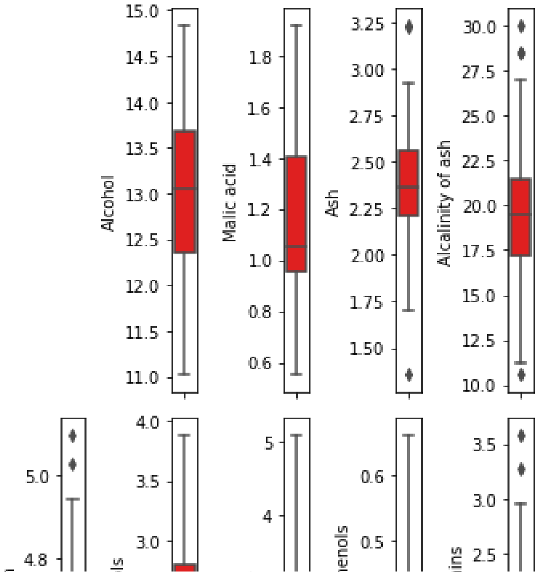
```python
wine.skew()
```

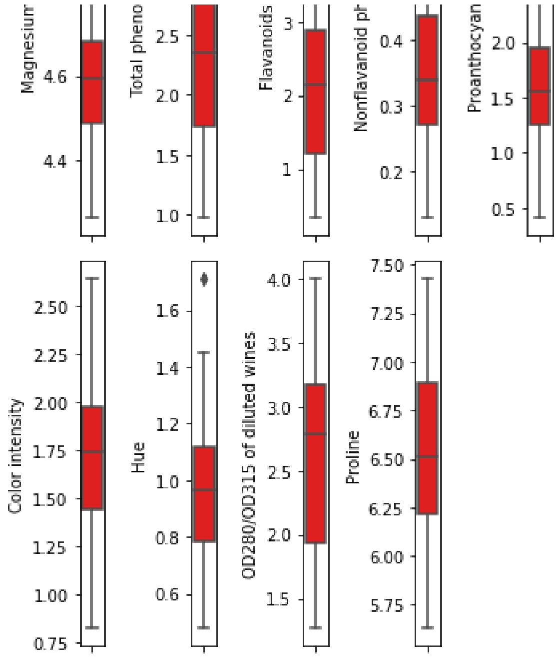
Out[16]:

```
Class                           0.107431
Alcohol                        -0.051482
Malic acid                      0.529222
Ash                            -0.176699
Alcalinity of ash               0.213047
Magnesium                       0.605723
Total phenols                   0.086639
Flavanoids                      0.025344
Nonflavanoid phenols            0.450151
Proanthocyanins                 0.517137
Color intensity                 0.097222
Hue                             0.021091
OD280/OD315 of diluted wines   -0.307285
Proline                         0.087930
dtype: float64
```

In [17]:

```python
col=wine.columns.values
ncol=5
nrow=7
plt.figure(figsize=(ncol,5*ncol))
for i in  range(1,len(col)):
    plt.subplot(nrow,ncol,i+1)
    sns.boxplot(wine[col[i]],color='red',orient='v')
    plt.tight_layout()
```





In [18]:

```
#Removing outliers
from scipy.stats import zscore
z_score=abs(zscore(wine))
print(wine.shape)
wine_df=wine.loc[(z_score<3).all(axis=1)]
print(wine_df.shape)
```

```
(178, 14)
(170, 14)
```

In [19]:

```
wine_df
```

Out[19]:

	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD28
0	1	14.23	0.996949	2.43	15.6	4.852030	2.80	3.06	0.28	2.29	1.893112	1.04	
1	1	13.20	1.022451	2.14	11.2	4.615121	2.65	2.76	0.26	1.28	1.682688	1.05	
2	1	13.16	1.211941	2.67	18.6	4.624973	2.80	3.24	0.30	2.81	1.899118	1.03	
3	1	14.37	1.081805	2.50	16.8	4.736198	3.85	3.49	0.24	2.18	2.174752	0.86	
4	1	13.24	1.278152	2.87	21.0	4.779123	2.80	2.69	0.39	1.82	1.671473	1.04	
...	...	...	...	...	...	...	...	...	...	...	...	...	
173	3	13.71	1.894617	2.45	20.5	4.564348	1.68	0.61	0.52	1.06	2.163323	0.64	
174	3	13.40	1.591274	2.48	23.0	4.634729	1.80	0.75	0.43	1.41	2.116256	0.70	
175	3	13.27	1.663926	2.26	20.0	4.795791	1.59	0.69	0.43	1.35	2.415914	0.59	
176	3	13.17	1.278152	2.37	20.0	4.795791	1.65	0.68	0.53	1.46	2.332144	0.60	
177	3	14.13	1.629241	2.74	24.5	4.574711	2.05	0.76	0.56	1.35	2.322388	0.61	

170 rows × 14 columns

In [20]:

```
wine_df=pd.DataFrame(data=wine_df)
```

In [21]:

```
wine_df
```

Out[21]:

	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD28
0	1	14.23	0.996949	2.43	15.6	4.852030	2.80	3.06	0.28	2.29	1.893112	1.04	
1	1	13.20	1.022451	2.14	11.2	4.615121	2.65	2.76	0.26	1.28	1.682688	1.05	
2	1	13.16	1.211941	2.67	18.6	4.624973	2.80	3.24	0.30	2.81	1.899118	1.03	
3	1	14.37	1.081805	2.50	16.8	4.736198	3.85	3.49	0.24	2.18	2.174752	0.86	
4	1	13.24	1.278152	2.87	21.0	4.779123	2.80	2.69	0.39	1.82	1.671473	1.04	
...	...	...	...	...	...	...	...	...	...	...	...	...	
173	3	13.71	1.894617	2.45	20.5	4.564348	1.68	0.61	0.52	1.06	2.163323	0.64	
174	3	13.40	1.591274	2.48	23.0	4.634729	1.80	0.75	0.43	1.41	2.116256	0.70	
175	3	13.27	1.663926	2.26	20.0	4.795791	1.59	0.69	0.43	1.35	2.415914	0.59	
176	3	13.17	1.278152	2.37	20.0	4.795791	1.65	0.68	0.53	1.46	2.332144	0.60	
177	3	14.13	1.629241	2.74	24.5	4.574711	2.05	0.76	0.56	1.35	2.322388	0.61	

170 rows × 14 columns

In [22]:

```
#x and y values allocation for training and testing
x=wine_df.iloc[:,1:-1]
```

In [23]:

```
x
```

Out[23]:

	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD280/OD31 of dilute wine
0	14.23	0.996949	2.43	15.6	4.852030	2.80	3.06	0.28	2.29	1.893112	1.04	3.9
1	13.20	1.022451	2.14	11.2	4.615121	2.65	2.76	0.26	1.28	1.682688	1.05	3.4
2	13.16	1.211941	2.67	18.6	4.624973	2.80	3.24	0.30	2.81	1.899118	1.03	3.1
3	14.37	1.081805	2.50	16.8	4.736198	3.85	3.49	0.24	2.18	2.174752	0.86	3.4
4	13.24	1.278152	2.87	21.0	4.779123	2.80	2.69	0.39	1.82	1.671473	1.04	2.9
...	...	...	...	...	...	...	...	...	...	...	...	.
173	13.71	1.894617	2.45	20.5	4.564348	1.68	0.61	0.52	1.06	2.163323	0.64	1.7
174	13.40	1.591274	2.48	23.0	4.634729	1.80	0.75	0.43	1.41	2.116256	0.70	1.5
175	13.27	1.663926	2.26	20.0	4.795791	1.59	0.69	0.43	1.35	2.415914	0.59	1.5
176	13.17	1.278152	2.37	20.0	4.795791	1.65	0.68	0.53	1.46	2.332144	0.60	1.6
177	14.13	1.629241	2.74	24.5	4.574711	2.05	0.76	0.56	1.35	2.322388	0.61	1.6

170 rows × 12 columns

In [24]:

```
x.shape
```

Out[24]:

```
(170, 12)
```

In [25]:

```
y=wine_df.iloc[:,0]
```

In [26]:

```
y
```

Out[26]:

```
0      1
1      1
2      1
3      1
4      1
      ..
173    3
174    3
175    3
176    3
177    3
Name: Class, Length: 170, dtype: int64
```

In [27]:

```
y.shape
```

Out[27]:

```
(170,)
```

In [28]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.22,random_state=42)
```

In [29]:

```
#we are using follwing models for prediction
#KNeighborsClassifier
#GaussianNB
#SVC
#DecisionTreeClassifier
#RandomForestClassifier
#AdaBoostClassifier
```

In [30]:

```
knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
knn.score(x_train,y_train)
predknn=knn.predict(x_test)
print(accuracy_score(y_test,predknn))
print(confusion_matrix(y_test,predknn))
print(classification_report(y_test,predknn))
```

```
0.9736842105263158
[[16  0  0]
 [ 0 13  1]
 [ 0  0  8]]
              precision    recall  f1-score   support

           1       1.00      1.00      1.00        16
           2       1.00      0.93      0.96        14
           3       0.89      1.00      0.94         8

    accuracy                           0.97        38
   macro avg       0.96      0.98      0.97        38
weighted avg       0.98      0.97      0.97        38
```

In [31]:

```
from sklearn.model_selection import cross_val_score
knnscores=cross_val_score(knn,x,y,cv=5)
print(knnscores)
print(knnscores.mean(),knnscores.std())
```

```
[0.88235294 0.94117647 0.88235294 1.         0.94117647]
0.9294117647058824 0.04401949866792873
```

In [32]:

```
mnb=MultinomialNB()
mnb.fit(x_train,y_train)
mnb.score(x_train,y_train)
predmnb=mnb.predict(x_test)
print(accuracy_score(y_test,predmnb))
print(confusion_matrix(y_test,predmnb))
print(classification_report(y_test,predmnb))
```

```
0.8947368421052632
[[13  3  0]
 [ 1 13  0]
 [ 0  0  8]]
              precision    recall  f1-score   support

           1       0.93      0.81      0.87        16
           2       0.81      0.93      0.87        14
           3       1.00      1.00      1.00         8

    accuracy                           0.89        38
   macro avg       0.91      0.91      0.91        38
weighted avg       0.90      0.89      0.89        38
```

In [33]:

```
mnbscores=cross_val_score(mnb,x,y,cv=5)
print(mnbscores)
print(mnbscores.mean(),mnbscores.std())
```

```
[0.85294118 0.91176471 0.73529412 0.91176471 1.        ]
0.8823529411764707 0.08724939396583131
```

In [34]:

```
svc=SVC(kernel='rbf')
svc.fit(x_train,y_train)
svc.score(x_train,y_train)
predsvc=svc.predict(x_test)
print(accuracy_score(y_test,predsvc))
print(confusion_matrix(y_test,predsvc))
print(classification_report(y_test,predsvc))
```

```
0.8947368421052632
[[14  2  0]
 [ 0 12  2]
 [ 0  0  8]]
              precision    recall  f1-score   support

           1       1.00      0.88      0.93        16
           2       0.86      0.86      0.86        14
           3       0.80      1.00      0.89         8

    accuracy                           0.89        38
   macro avg       0.89      0.91      0.89        38
weighted avg       0.91      0.89      0.90        38
```

In [35]:

```
svcscores=cross_val_score(svc,x,y,cv=5)
print(svcscores)
print(svcscores.mean(),svcscores.std())
```

```
[0.70588235 0.79411765 0.73529412 0.97058824 0.88235294]
0.8176470588235294 0.09737026680733438
```

In [36]:

```
dtc=DecisionTreeClassifier()
dtc.fit(x_train,y_train)
dtc.score(x_train,y_train)
preddtc=dtc.predict(x_test)
print(accuracy_score(y_test,preddtc))
print(confusion_matrix(y_test,preddtc))
print(classification_report(y_test,preddtc))
```

```
1.0
[[16  0  0]
 [ 0 14  0]
 [ 0  0  8]]
              precision    recall  f1-score   support

           1       1.00      1.00      1.00        16
           2       1.00      1.00      1.00        14
           3       1.00      1.00      1.00         8

    accuracy                           1.00        38
   macro avg       1.00      1.00      1.00        38
weighted avg       1.00      1.00      1.00        38
```

In [37]:

```
dtcscores=cross_val_score(dtc,x,y,cv=5)
print(dtcscores)
print(dtcscores.mean(),dtcscores.std())
```

```
[0.88235294 0.82352941 0.91176471 0.97058824 1.        ]
0.9176470588235294 0.06280634265900772
```

In [38]:

```
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
rf.score(x_train,y_train)
predrf=rf.predict(x_test)
print(accuracy_score(y_test,predrf))
print(confusion_matrix(y_test,predrf))
print(classification_report(y_test,predrf))
```

```
0.9736842105263158
[[16  0  0]
 [ 0 13  1]
 [ 0  0  8]]
              precision    recall  f1-score   support

           1       1.00      1.00      1.00        16
           2       1.00      0.93      0.96        14
           3       0.89      1.00      0.94         8

    accuracy                           0.97        38
   macro avg       0.96      0.98      0.97        38
weighted avg       0.98      0.97      0.97        38
```

In [39]:

```
rfscores=cross_val_score(rf,x,y,cv=5)
print(rfscores)
print(rfscores.mean(),rfscores.std())
```

```
[0.94117647 0.97058824 1.         1.         1.        ]
0.9823529411764707 0.023529411764705895
```

In [40]:

```python
ad=AdaBoostClassifier()
ad.fit(x_train,y_train)
ad.score(x_train,y_train)
predad=ad.predict(x_test)
print(accuracy_score(y_test,predad))
print(confusion_matrix(y_test,predad))
print(classification_report(y_test,predad))
```

```
0.5789473684210527
[[ 0 16  0]
 [ 0 14  0]
 [ 0  0  8]]
              precision    recall  f1-score   support

           1       0.00      0.00      0.00        16
           2       0.47      1.00      0.64        14
           3       1.00      1.00      1.00         8

    accuracy                           0.58        38
   macro avg       0.49      0.67      0.55        38
weighted avg       0.38      0.58      0.44        38
```

In [41]:

```python
adscores=cross_val_score(ad,x,y,cv=5)
print(adscores)
print(adscores.mean(),adscores.std())
```

```
[0.70588235 1.         0.91176471 0.91176471 0.88235294]
0.8823529411764707 0.09665692191267634
```

In [42]:

```python
#RandomForestClassifier and DecisionTreeClassifier are the best model among all models
import joblib
joblib.dump(dtc,'wine_df.pkl')
```

Out[42]:

```
['wine_df.pkl']
```

In [ ]:

```python

```