In [1]:
```
import pandas as pd
data_wine = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-database
```

In [2]:
```
data_wine.describe()
```

Out[2]:

	Cultivar	A	B	C	D	E	F	
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.00
mean	1.938202	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.02
std	0.775035	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.99
min	1.000000	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.34
25%	1.000000	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.20
50%	2.000000	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.13
75%	3.000000	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.87
max	3.000000	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.08

In [4]:
```
data_wine.shape
```

Out[4]:
```
(178, 14)
```

In [5]:
```
# Classify the data
x = data_wine.drop('Cultivar',axis=1)
y = data_wine['Cultivar']
```

In [6]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y)
```

In [7]:
```
#data preprocessing
from sklearn.preprocessing import StandardScaler
scale = StandardScaler()
```

In [8]:
```
scale.fit(x_train)
```

Out[8]:
```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

In [9]:
```
x_train = scale.transform(x_train)
x_test = scale.transform(x_test)
```

In [10]:

```
from sklearn.neural_network import MLPClassifier
mlp = MLPClassifier(hidden_layer_sizes=(12,13,14),max_iter=500)
mlp.fit(x_train,y_train)
```

Out[10]:

```
MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
       beta_2=0.999, early_stopping=False, epsilon=1e-08,
       hidden_layer_sizes=(12, 13, 14), learning_rate='constant',
       learning_rate_init=0.001, max_iter=500, momentum=0.9,
       nesterovs_momentum=True, power_t=0.5, random_state=None,
       shuffle=True, solver='adam', tol=0.0001, validation_fraction=0.1,
       verbose=False, warm_start=False)
```

In [11]:

```
clf_predict = mlp.predict(x_test)
```

In [13]:

```
from sklearn.metrics import classification_report,confusion_matrix
print(confusion_matrix(y_test,clf_predict))
```

```
[[15  0  0]
 [ 0 22  1]
 [ 0  0  7]]
```

In [14]:

```
print(classification_report(y_test,clf_predict))
```

```
             precision    recall  f1-score   support

          1       1.00      1.00      1.00        15
          2       1.00      0.96      0.98        23
          3       0.88      1.00      0.93         7

avg / total       0.98      0.98      0.98        45
```

In [ ]:

```

```