In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn import  metrics
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import KFold, LeaveOneOut, ShuffleSplit, StratifiedKFold
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score, confusion_matrix
```

In [2]:
```
iris = pd.read_csv('D:\\24 - Machine_Learning\\download files\\iris.csv')
iris
```

Out[2]:

	Sepal length	Sepal width	Petal length	Petal width	Class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [3]:
```
iris.dtypes
```

Out[3]:
```
Sepal length    float64
Sepal width     float64
Petal length    float64
Petal width     float64
Class            object
dtype: object
```

In [4]:
```
features = iris[['Sepal length', 'Sepal width', 'Petal length', 'Petal width']].values
classes = iris['Class'].values
```

In [5]:
```
# k-fold cross validation technique
kf = KFold(n_splits=5, random_state=1,shuffle=True)
# stratified kfold cross validation technique
skf = StratifiedKFold(n_splits=5)
# LeaveOneOut cross validation technique
loocv = LeaveOneOut()
# shuffle split cross validation technique
shvc = ShuffleSplit()
```

In [6]:
```
# evaluating the data sets with kfold and DecisionTreeClassifier
dst = DecisionTreeClassifier()
scores = cross_val_score(dst,features,classes,scoring='accuracy',cv=kf)
print('Accuracy using Decision Tree: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using Decision Tree: 94.000000%
[0.96666667 0.96666667 0.96666667 0.93333333 0.86666667]
```

In [7]:
```
# evaluating the data sets with kfold and Naive Bayes Classifier
nb = GaussianNB()
scores = cross_val_score(nb,features,classes,scoring='accuracy',cv=kf)
print('Accuracy using GaussianNB: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using GaussianNB: 95.333333%
[0.96666667 0.96666667 1.         0.9        0.93333333]
```

In [8]:
```
# evaluating the data sets with kfold and SVM
svm = SVC()
scores = cross_val_score(svm,features,classes,scoring='accuracy',cv=kf)
print('Accuracy using SVC: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using SVC: 94.666667%
[0.96666667 0.93333333 0.9        0.96666667 0.96666667]
```

In [9]:
```
# evaluating the data sets with kfold and KNN Classifier
```

```
knn = KNeighborsClassifier()
scores = cross_val_score(knn,features,classes,scoring='accuracy',cv=kf)
print('Accuracy using KNN: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using KNN: 95.333333%
[1.         0.96666667 0.9        0.93333333 0.96666667]
```

In [10]:
```
# evaluating the data sets with Stratified KFold and DecisionTree
scores = cross_val_score(dst,features,classes,scoring='accuracy',cv=skf)
print('Accuracy using Decision Tree: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using Decision Tree: 96.666667%
[0.96666667 0.96666667 0.9        1.         1.        ]
```

In [11]:
```
# evaluating the data sets Stratified KFold and Naive Bayes Classifier
scores = cross_val_score(nb,features,classes,scoring='accuracy',cv=skf)
print('Accuracy using GaussianNB: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using GaussianNB: 95.333333%
[0.93333333 0.96666667 0.93333333 0.93333333 1.        ]
```

In [12]:
```
# evaluating the data sets Stratified KFold and SVM
scores = cross_val_score(svm,features,classes,scoring='accuracy',cv=skf)
print('Accuracy using SVM: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using SVM: 96.666667%
[0.96666667 0.96666667 0.96666667 0.93333333 1.        ]
```

In [13]:
```
# evaluating the data sets Stratified KFold and KNN
scores = cross_val_score(knn,features,classes,scoring='accuracy',cv=skf)
print('Accuracy using KNN: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using KNN: 97.333333%
[0.96666667 1.         0.93333333 0.96666667 1.        ]
```

In [14]:
```
# evaluating the data sets with LeaveOneOut and DecisionTree
scores = cross_val_score(dst,features,classes,scoring='accuracy',cv=loocv)
print('Accuracy using GaussianNB: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using GaussianNB: 94.000000%
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1.
 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.]
```

In [15]:
```
# evaluating the data sets LeaveOneOut and Naive Bayes Classifier
scores = cross_val_score(nb,features,classes,scoring='accuracy',cv=loocv)
print('Accuracy using GaussianNB: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using GaussianNB: 95.333333%
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1.
 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.]
```

In [16]:
```
# evaluating the data sets LeaveOneOut and SVM
scores = cross_val_score(svm,features,classes,scoring='accuracy',cv=loocv)
print('Accuracy using SVM: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using SVM: 96.666667%
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.]
```

In [17]:
```
# evaluating the data sets LeaveOneOut and KNN
scores = cross_val_score(knn,features,classes,scoring='accuracy',cv=loocv)
print('Accuracy using KNN: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using KNN: 96.666667%
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1.
 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.]
```

In [18]:
```
# evaluating the data sets with ShuffleSplit and DecisionTree
scores = cross_val_score(dst,features,classes,scoring='accuracy',cv=shvc)
print('Accuracy using GaussianNB: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using GaussianNB: 94.666667%
[1.         1.         0.93333333 0.86666667 1.         0.86666667
 0.86666667 1.         1.         0.93333333]
```

In [19]:
```
# evaluating the data sets with ShuffleSplit and Naive Bayes Classifier
scores = cross_val_score(nb,features,classes,scoring='accuracy',cv=shvc)
print('Accuracy using GaussianNB: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using GaussianNB: 95.333333%
[1.         0.93333333 1.         0.8        0.86666667 1.
 0.93333333 1.         1.         1.        ]
```

In [20]:
```
# evaluating the data sets with ShuffleSplit and SVM
scores = cross_val_score(svm,features,classes,scoring='accuracy',cv=shvc)
print('Accuracy using GaussianNB: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using GaussianNB: 98.000000%
[1.         1.         0.93333333 0.93333333 1.         1.
 1.         1.         1.         0.93333333]
```

In [21]:
```
# evaluating the data sets with ShuffleSplit and KNN
scores = cross_val_score(knn,features,classes,scoring='accuracy',cv=shvc)
print('Accuracy using KNN: %2f%%'%(scores.mean()*100))
print(scores)
```

```
Accuracy using KNN: 96.000000%
[1.         0.86666667 1.         1.         1.         0.86666667
 1.         0.93333333 1.         0.93333333]
```