In [4]:
```
#Voting Classifier
```

In [5]:
```
#import libraries
import pandas as pd
from sklearn import datasets

#load dataset
df = datasets.load_iris()

#independent variables
X = df['data']

#dependent variable
y = df['target']
```

In [6]:
```
#split the dataset into train and test data
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test=train_test_split(X,y,test_size=0.2,random_state=0)
```

In [7]:
```
#import Sklearn libraries

from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier
from sklearn.metrics import accuracy_score
```

In [8]:
```
#creating objects for the algorithms

svc = SVC(probability=True)
knn = KNeighborsClassifier()
dt = DecisionTreeClassifier()
```

In [9]:
```
#performing voting classifier with hard voting (probablity or majority)

hard_vote = VotingClassifier(estimators=[('SVC', svc), ('KNNC', knn), ('DTC', dt)], voting='hard')
hard_vote.fit(X_train, y_train)
y_pred = hard_vote.predict(X_test)

#accuracy score (actual value-predict value)
score = accuracy_score(y_test, y_pred)
print('accuracy score:',score)
```

```
accuracy score: 1.0
```

In [10]:
```
#reporting the classification metrics

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

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11
           1       1.00      1.00      1.00        13
           2       1.00      1.00      1.00         6

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

In [11]:
```
#performing voting classifier with soft voting (aggregating)

soft_vote = VotingClassifier(estimators=[('SVC', svc), ('KNNC', knn), ('DTC', dt)], voting='soft')
soft_vote.fit(X_train, y_train)
y_pred = soft_vote.predict(X_test)

#accuracy score
score = accuracy_score(y_pred, y_test)
print('accuracy score:',score)
```

```
accuracy score: 1.0
```

In [29]:
```
#reporting the classification metrics

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

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11
           1       1.00      1.00      1.00        13
           2       1.00      1.00      1.00         6

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

In [37]:
```
#Cross Validation

from  sklearn.model_selection import KFold

#perform K-fold cross-validation
cv = KFold(n_splits=10, random_state=1, shuffle=True)
```

In [35]:
```
from sklearn.model_selection import cross_val_score

labels = ['Support Vector Classifier', 'KNN Classifier', 'Decision Tree Classifier']

for classifier, label in zip([svc, knn, dt], labels):
    scores = cross_val_score(classifier, X_test, y_test, cv=cv, scoring='accuracy')
    print('accuracy: %0.2f (+/- %0.2f) [%s]'%(scores.mean(), scores.std(), label))
```

```
accuracy: 0.80 (+/- 0.07) [Support Vector Classifier]
accuracy: 0.93 (+/- 0.08) [KNN Classifier]
accuracy: 0.87 (+/- 0.07) [Decision Tree Classifier]
```

In [36]:
```
from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11
           1       1.00      1.00      1.00        13
           2       1.00      1.00      1.00         6

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```