Apply Bagging , Boosting and Stacking on Iris Dataset

In [4]:

```
from sklearn.datasets import load_iris
dataset = load_iris()
dataset
```

```
R.A. "The use of multiple measurements in taxonomic problems"\n     Annual
Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to\n     Math
ematical Statistics" (John Wiley, NY, 1950).\n   - Duda, R.O., & Hart, P.
E. (1973) Pattern Classification and Scene Analysis.\n     (Q327.D83) John
Wiley & Sons.  ISBN 0-471-22361-1.  See page 218.\n   - Dasarathy, B.V. (1
980) "Nosing Around the Neighborhood: A New System\n     Structure and Cla
ssification Rule for Recognition in Partially Exposed\n     Environments".
IEEE Transactions on Pattern Analysis and Machine\n     Intelligence, Vol.
PAMI-2, No. 1, 67-71.\n   - Gates, G.W. (1972) "The Reduced Nearest Neighb
or Rule".  IEEE Transactions\n     on Information Theory, May 1972, 431-43
3.\n   - See also: 1988 MLC Proceedings, 54-64.  Cheeseman et al"s AUTOCLA
SS II\n     conceptual clustering system finds 3 classes in the data.\n
- Many, many more ...',
 'feature_names': ['sepal length (cm)',
  'sepal width (cm)',
  'petal length (cm)',
  'petal width (cm)'],
 'filename': 'C:\\ProgramData\\Anaconda3\\lib\\site-packages\\sklearn\\dat
asets\\data\\iris.csv'}
```

In [6]:

```
x = dataset.data
y = dataset.target
```

In [7]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=1)
```

In [8]:

```
print('Shape of x_train is: ',x_train.shape)
print('Shape of x_test is: ',x_test.shape)
print('Shape of y_train is: ',y_train.shape)
print('Shape of y_train is: ',y_test.shape)
```

```
Shape of x_train is:  (120, 4)
Shape of x_test is:  (30, 4)
Shape of y_train is:  (120,)
Shape of y_train is:  (30,)
```

In [9]:

```
from sklearn.ensemble import BaggingClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score
```

In [13]:

```
dtc = DecisionTreeClassifier()
model = BaggingClassifier(base_estimator=dtc,n_estimators= 100 ,random_state= 42)
results = cross_val_score(model, x,y, cv= 10)

print(results.mean())
```

```
0.96
```

AdaBoost Classification

In [14]:

```
# AdaBoost Classification
from sklearn.ensemble import AdaBoostClassifier
model = AdaBoostClassifier(n_estimators=100, random_state= 42)
results = cross_val_score(model, x,y, cv= 10)
print(results.mean())
```

```
0.9466666666666667
```

Stacking

In [17]:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.ensemble import VotingClassifier
from sklearn.naive_bayes import GaussianNB

# create the sub models

estimators = []
model1 = GaussianNB()
estimators.append(('Naive_Bais', model1))
model2 = DecisionTreeClassifier()
estimators.append(('cart', model2))
model3 = SVC()
estimators.append(('svm', model3))

# create the ensemble model
ensemble = VotingClassifier(estimators)
results = cross_val_score(ensemble, x,y ,cv= 10)
print(results.mean())
```

```
0.9666666666666666
```

In [18]:

```
ensemble
```

Out[18]:

```
VotingClassifier(estimators=[('Naive_Bais', GaussianNB()),
                             ('cart', DecisionTreeClassifier()),
                             ('svm', SVC())])
```

In [19]:

```
results
```

Out[19]:

```
array([1.        , 0.93333333, 1.        , 0.93333333, 0.93333333,
       0.93333333, 0.93333333, 1.        , 1.        , 1.        ])
```

GradientBoostingClassifier

In [20]:

```
# importing machine learning models for prediction
from sklearn.ensemble import GradientBoostingClassifier
```

In [22]:

```
# initializing the boosting module with default parameters
model = GradientBoostingClassifier()
#model = AdaBoostClassifier(n_estimators=100, random_state= 42)
results = cross_val_score(model, x,y, cv= 10)
print(results.mean())
```

```
0.96
```

In [ ]:

```

```