In [10]:
```
#import libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from scipy.spatial import distance
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier

#read data file as dataframe
data = pd.read_csv('iris.csv')

#take four numeric features as X inputy
X = data.values[:, :4]

#create an array of length 150 named y
y = np.zeros(150)

#encoding classes to numbers
for i in range(len(y)):
    if data.values[i, 4]=='setosa':
        y[i] = 0
    elif data.values[i, 4]=='versicolor':
        y[i] = 1
    elif data.values[i, 4]=='virginica':
        y[i] = 2

#randomly shuffle the whole dataset and create train-test partition
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_
distance_matrix=distance.cdist(X_test,X_train,'euclidean')
print(distance_matrix)
print(distance_matrix.shape)
print(distance.cdist(X_test,X_train).shape)
z=np.argsort(distance.cdist(X_test,X_train))
```

```
[[ 0.65574385  2.59036677  0.64807407 ...,  3.84447656  0.78740079
   1.81383571]
 [ 2.83901391  5.61248608  3.15119025 ...,  0.55677644  2.74772633
   4.84561658]
 [ 3.51852242  0.54772256  3.31209903 ...,  6.5169011   3.6373067
   1.2489996 ]
 ...,
 [ 1.57162336  1.6583124   1.3190906  ...,  4.7138095   1.79443584
   0.8660254 ]
 [ 0.93273791  2.51594913  1.00995049 ...,  3.64965752  1.          1.72916165]
 [ 2.79105715  0.96436508  2.54558441 ...,  5.60535458  3.00998339
   0.75498344]]
(50, 100)
(50, 100)
```

In [16]:
```
#
d = distance.cdist(X_test,X_train,'euclidean')
print(d.shape)


for i in range (50):
    yy=np.argsort(d,axis=1)
print(y)

k=int(13)
j=int(0)
y_pre=[]
for j in range(0,50):
    yyy = np.zeros(3)
    for i in range (0,k):
        ind=int(yy[j][i])
        val=int(y_train[ind])
        yyy[val]+=1
    print(np.argmax(yyy, axis=0))
    y_pre.append(np.argmax(yyy, axis=0))

print(y_pre)
accuracy_score(y_test, y_pre)
```

```
(50, 100)
[ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.
  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.
  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  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.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.
  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  2.  2.  2.  2.  2.  2.  2.  2.
  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.
  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.  2.
  2.  2.  2.  2.  2.  2.]
1
0
2
1
1
0
1
2
1
1
2
0
0
0
0
1
2
1
1
2
0
2
0
```

```
2
2
2
2
2
0
0
0
0
1
0
0
2
1
0
0
0
2
1
1
0
0
1
2
2
1
2
[1, 0, 2, 1, 1, 0, 1, 2, 1, 1, 2, 0, 0, 0, 0, 1, 2, 1, 1, 2, 0, 2, 0, 2, 2,
2, 2, 2, 0, 0, 0, 0, 1, 0, 0, 2, 1, 0, 0, 0, 2, 1, 1, 0, 0, 1, 2, 2, 1, 2]
```

Out[16]: 1.0

In [ ]: