MT18052_Test1

September 6, 2019

In [1]:
```
import pandas as pd
import csv
import numpy as np
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import matplotlib
from sklearn.manifold import TSNE
import warnings
warnings.filterwarnings("ignore")
# matplotlib.use('Agg')
```

In [2]:
```
data = []
labels = []
with open('iris.data') as csvfile:
    readCSV = csv.reader(csvfile, delimiter=',')
    for row in readCSV:
#         data.append(row[0:4])
        labels.append(row[4])
        tmp = np.array(row[0:4]).astype(float)
        data.append(tmp)
data = np.array(data)
labels = np.array (labels)
```

In [3]:
```
print (len(data))
```

```
150
```

In [4]:
```
def plotgraph(resultlist,title=""):
    '''
    fig = plt.figure()
    ax = fig.gca(projection='3d')
    '''
#     plt.figure()
    for i in sorted(resultlist.keys()):
        '''
        x = resultlist[i].T[0]
        y = resultlist[i].T[1]
```

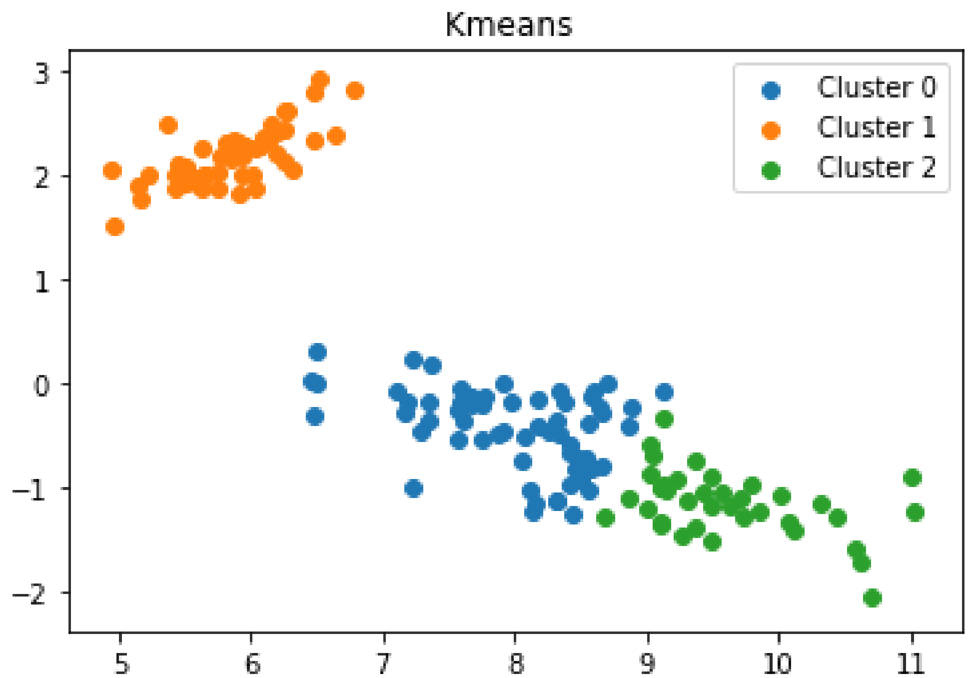
```
            z = resultlist[i].T[2]

            ax.scatter(x,y,z,label='Cluster ' + str(i))
            '''
            plt.scatter(resultlist[i].T[0],resultlist[i].T[1],label='Cluster ' + str(i))
            #plt.show()
        plt.legend(loc='best')
        plt.title(title)
        plt.show()
```

In [17]:
```
def completefun(datapoints,numclusters=3):
#     print (datapoints)
    kmeans = KMeans(n_clusters=numclusters, random_state=0).fit(datapoints)
    svd = TruncatedSVD(n_components=2)
    tnsepoints = svd.fit_transform(data)
    clusters = {}
    predicted_labels = []
    for i in range(len(kmeans.labels_)):
        predicted_labels.append(kmeans.labels_[i])
        if(kmeans.labels_[i] not in clusters):
            clusters[kmeans.labels_[i]] = []
        clusters[kmeans.labels_[i]].append(i)
    for i in clusters.keys():
        clusters[i] = tnsepoints[clusters[i]]
#     print (clusters)

    plotgraph(clusters,"Kmeans ")
    return kmeans.labels_
#     print ("Clusters using Kmeans: %d"%(len(set(kmeans.labels_))))
```

In [18]:
```
predicted_labels = completefun(data)
```




```
In [7]: print (predicted_labels)
```

```
[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 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2 2 2 2 0 2 2 2 2
 2 2 0 0 2 2 2 2 0 2 0 2 0 2 2 0 0 2 2 2 2 2 0 2 2 2 2 0 2 2 2 0 2 2 2 0 2
 2 0]
```

```
In [19]: print(labels)
```

```
['Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
```

```
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica']
```

In [8]:
```
image = data[0]
```

In [9]:
```
from matplotlib import pyplot as plt
from sklearn.decomposition import TruncatedSVD
```

In [10]:
```
svd = TruncatedSVD(n_components=2)
scd_data = svd.fit_transform(data)
```

In [11]:
```
type(data[0][0])
```

Out[11]:
```
numpy.float64
```

2 clusters are close to each other but are easily saperable but 1 cluster is fr from others

In [ ]: