

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 [34]: def completfun(datapoints,numclusters=3):
#     print (datapoints)
    kmeans = KMeans(n_clusters=numclusters, random_state=0).fit(datapoints)
    svd = TruncatedSVD(n_components=2)
    tnsepoints = svd.fit_transform(datapoints)
    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)

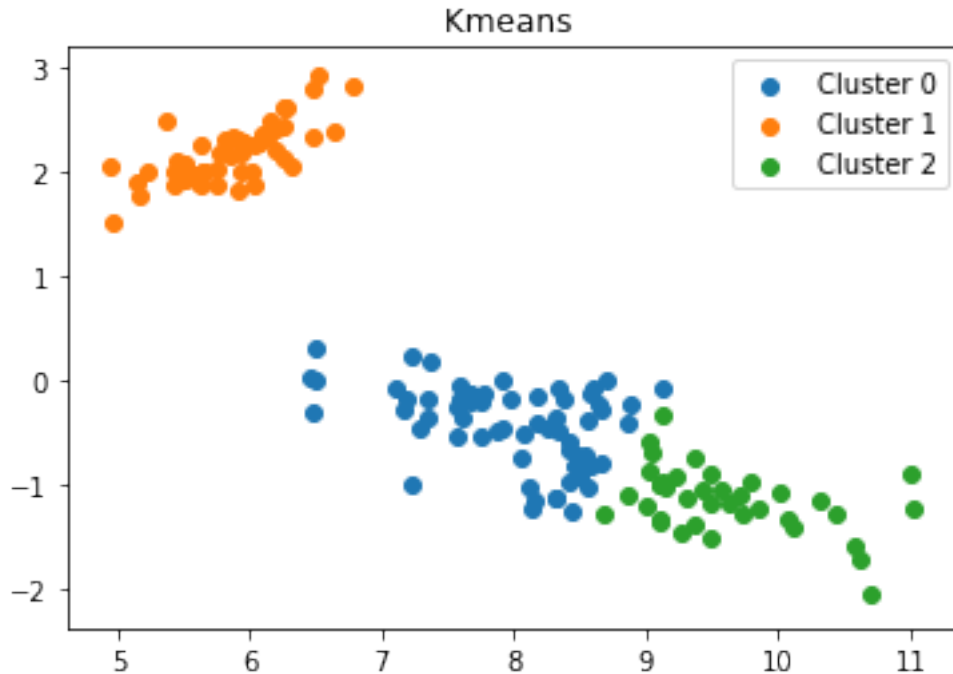
    plotgraph(clusters,"Kmeans ")
    return kmeans.labels_
#     print ("Clusters using Kmeans: %d"%(len(set(kmeans.labels_))))

```

```

In [35]: predicted_labels = completfun(data)

```



```
In [36]: 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 0 0 2 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 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 0]
```

```
In [37]: 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-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'

```

```
In [38]: image = data[0]
```

```
In [39]: from matplotlib import pyplot as plt
         from sklearn.decomposition import TruncatedSVD
```

```
In [40]: svd = TruncatedSVD(n_components=2)
         scd_data = svd.fit_transform(data)
```

```
In [41]: y = set(labels)
         clusters = {}
         data = np.array(data)
         for i in range(len(labels)):
             # predicted_labels.append(kmeans.labels_[i])
             if(labels[i] not in clusters):
                 clusters[labels[i]] = []
                 clusters[labels[i]].append(i)
         for i in clusters.keys():
             clusters[i] = data[clusters[i]]
```

```
In [42]: type(data[0][0])
```

```
Out[42]: numpy.float64
```

```
In [43]: clusters
```

```
Out[43]: {'Iris-setosa': array([[5.1, 3.5, 1.4, 0.2],
                                [4.9, 3. , 1.4, 0.2],
```

```

[4.7, 3.2, 1.3, 0.2],
[4.6, 3.1, 1.5, 0.2],
[5. , 3.6, 1.4, 0.2],
[5.4, 3.9, 1.7, 0.4],
[4.6, 3.4, 1.4, 0.3],
[5. , 3.4, 1.5, 0.2],
[4.4, 2.9, 1.4, 0.2],
[4.9, 3.1, 1.5, 0.1],
[5.4, 3.7, 1.5, 0.2],
[4.8, 3.4, 1.6, 0.2],
[4.8, 3. , 1.4, 0.1],
[4.3, 3. , 1.1, 0.1],
[5.8, 4. , 1.2, 0.2],
[5.7, 4.4, 1.5, 0.4],
[5.4, 3.9, 1.3, 0.4],
[5.1, 3.5, 1.4, 0.3],
[5.7, 3.8, 1.7, 0.3],
[5.1, 3.8, 1.5, 0.3],
[5.4, 3.4, 1.7, 0.2],
[5.1, 3.7, 1.5, 0.4],
[4.6, 3.6, 1. , 0.2],
[5.1, 3.3, 1.7, 0.5],
[4.8, 3.4, 1.9, 0.2],
[5. , 3. , 1.6, 0.2],
[5. , 3.4, 1.6, 0.4],
[5.2, 3.5, 1.5, 0.2],
[5.2, 3.4, 1.4, 0.2],
[4.7, 3.2, 1.6, 0.2],
[4.8, 3.1, 1.6, 0.2],
[5.4, 3.4, 1.5, 0.4],
[5.2, 4.1, 1.5, 0.1],
[5.5, 4.2, 1.4, 0.2],
[4.9, 3.1, 1.5, 0.1],
[5. , 3.2, 1.2, 0.2],
[5.5, 3.5, 1.3, 0.2],
[4.9, 3.1, 1.5, 0.1],
[4.4, 3. , 1.3, 0.2],
[5.1, 3.4, 1.5, 0.2],
[5. , 3.5, 1.3, 0.3],
[4.5, 2.3, 1.3, 0.3],
[4.4, 3.2, 1.3, 0.2],
[5. , 3.5, 1.6, 0.6],
[5.1, 3.8, 1.9, 0.4],
[4.8, 3. , 1.4, 0.3],
[5.1, 3.8, 1.6, 0.2],
[4.6, 3.2, 1.4, 0.2],
[5.3, 3.7, 1.5, 0.2],
[5. , 3.3, 1.4, 0.2]]), 'Iris-versicolor': array([[7. , 3.2, 4.7, 1.4],

```

[6.4, 3.2, 4.5, 1.5],
 [6.9, 3.1, 4.9, 1.5],
 [5.5, 2.3, 4. , 1.3],
 [6.5, 2.8, 4.6, 1.5],
 [5.7, 2.8, 4.5, 1.3],
 [6.3, 3.3, 4.7, 1.6],
 [4.9, 2.4, 3.3, 1.],
 [6.6, 2.9, 4.6, 1.3],
 [5.2, 2.7, 3.9, 1.4],
 [5. , 2. , 3.5, 1.],
 [5.9, 3. , 4.2, 1.5],
 [6. , 2.2, 4. , 1.],
 [6.1, 2.9, 4.7, 1.4],
 [5.6, 2.9, 3.6, 1.3],
 [6.7, 3.1, 4.4, 1.4],
 [5.6, 3. , 4.5, 1.5],
 [5.8, 2.7, 4.1, 1.],
 [6.2, 2.2, 4.5, 1.5],
 [5.6, 2.5, 3.9, 1.1],
 [5.9, 3.2, 4.8, 1.8],
 [6.1, 2.8, 4. , 1.3],
 [6.3, 2.5, 4.9, 1.5],
 [6.1, 2.8, 4.7, 1.2],
 [6.4, 2.9, 4.3, 1.3],
 [6.6, 3. , 4.4, 1.4],
 [6.8, 2.8, 4.8, 1.4],
 [6.7, 3. , 5. , 1.7],
 [6. , 2.9, 4.5, 1.5],
 [5.7, 2.6, 3.5, 1.],
 [5.5, 2.4, 3.8, 1.1],
 [5.5, 2.4, 3.7, 1.],
 [5.8, 2.7, 3.9, 1.2],
 [6. , 2.7, 5.1, 1.6],
 [5.4, 3. , 4.5, 1.5],
 [6. , 3.4, 4.5, 1.6],
 [6.7, 3.1, 4.7, 1.5],
 [6.3, 2.3, 4.4, 1.3],
 [5.6, 3. , 4.1, 1.3],
 [5.5, 2.5, 4. , 1.3],
 [5.5, 2.6, 4.4, 1.2],
 [6.1, 3. , 4.6, 1.4],
 [5.8, 2.6, 4. , 1.2],
 [5. , 2.3, 3.3, 1.],
 [5.6, 2.7, 4.2, 1.3],
 [5.7, 3. , 4.2, 1.2],
 [5.7, 2.9, 4.2, 1.3],
 [6.2, 2.9, 4.3, 1.3],
 [5.1, 2.5, 3. , 1.1],

```

[5.7, 2.8, 4.1, 1.3]]), 'Iris-virginica': array([[6.3, 3.3, 6. , 2.5],
[5.8, 2.7, 5.1, 1.9],
[7.1, 3. , 5.9, 2.1],
[6.3, 2.9, 5.6, 1.8],
[6.5, 3. , 5.8, 2.2],
[7.6, 3. , 6.6, 2.1],
[4.9, 2.5, 4.5, 1.7],
[7.3, 2.9, 6.3, 1.8],
[6.7, 2.5, 5.8, 1.8],
[7.2, 3.6, 6.1, 2.5],
[6.5, 3.2, 5.1, 2. ],
[6.4, 2.7, 5.3, 1.9],
[6.8, 3. , 5.5, 2.1],
[5.7, 2.5, 5. , 2. ],
[5.8, 2.8, 5.1, 2.4],
[6.4, 3.2, 5.3, 2.3],
[6.5, 3. , 5.5, 1.8],
[7.7, 3.8, 6.7, 2.2],
[7.7, 2.6, 6.9, 2.3],
[6. , 2.2, 5. , 1.5],
[6.9, 3.2, 5.7, 2.3],
[5.6, 2.8, 4.9, 2. ],
[7.7, 2.8, 6.7, 2. ],
[6.3, 2.7, 4.9, 1.8],
[6.7, 3.3, 5.7, 2.1],
[7.2, 3.2, 6. , 1.8],
[6.2, 2.8, 4.8, 1.8],
[6.1, 3. , 4.9, 1.8],
[6.4, 2.8, 5.6, 2.1],
[7.2, 3. , 5.8, 1.6],
[7.4, 2.8, 6.1, 1.9],
[7.9, 3.8, 6.4, 2. ],
[6.4, 2.8, 5.6, 2.2],
[6.3, 2.8, 5.1, 1.5],
[6.1, 2.6, 5.6, 1.4],
[7.7, 3. , 6.1, 2.3],
[6.3, 3.4, 5.6, 2.4],
[6.4, 3.1, 5.5, 1.8],
[6. , 3. , 4.8, 1.8],
[6.9, 3.1, 5.4, 2.1],
[6.7, 3.1, 5.6, 2.4],
[6.9, 3.1, 5.1, 2.3],
[5.8, 2.7, 5.1, 1.9],
[6.8, 3.2, 5.9, 2.3],
[6.7, 3.3, 5.7, 2.5],
[6.7, 3. , 5.2, 2.3],
[6.3, 2.5, 5. , 1.9],
[6.5, 3. , 5.2, 2. ],

```

```
[6.2, 3.4, 5.4, 2.3],  
[5.9, 3. , 5.1, 1.8]]))}
```

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

```
In [44]: svd = TruncatedSVD(n_components=2)  
         tnsepoints = svd.fit_transform(data)  
         plotgraph(clusters,"Original ")
```



```
In [ ]:
```