1. Рубежный контроль № 2

2. Вариант №2. Кластеризация данных.

Мельников К.И. ИУ5-24М

Необходимо решить задачу кластеризации на основе любого выбранного Вами датасета.

Кластеризуйте данные с помощью трех различных алгоритмов кластеризации. Алгоритмы выбираются произвольным образом, рекомендуется использовать алгоритмы из лекции.

Сравните качество кластеризации для трех алгоритмов с помощью следующих метрик качества кластеризации:

```
Adjusted Rand index
Adjusted Mutual Information
Homogeneity, completeness, V-measure
Коэффициент силуэта
```

3. Ход работы

```
In [2]: import numpy as np

        from PIL  import Image

        from sklearn.decomposition import PCA
        from sklearn.preprocessing import scale
        from sklearn import metrics
        from sklearn.cluster import KMeans

        from IPython.display import display

        from tqdm import tqdm_notebook as tqdm

        import matplotlib.pyplot as plt
        import matplotlib.cm as cm

        from scipy import ndimage as ndi
        from skimage.morphology import medial_axis

        from scipy import ndimage as ndi
        from skimage.morphology import medial_axis
        from skimage.morphology import skeletonize

        from scipy.spatial import Delaunay

        from sklearn.model_selection import train_test_split
```

```
In [7]: data = np.load("hiragana.npz")['arr_0']
```

```
In [8]: X = []
        Y = []
        for index,letter in enumerate(data):
            for variant in letter:
                X.append(variant)
                Y.append(index)
```

```
In [38]: def moment(array,m1,m2):
             moment = 0
             for y,ver in enumerate(array):
                 for x,hor in enumerate(ver):
                     moment += pow(x,m1)*pow(y,m2)*hor
             return moment

         def center(array):
             x = moment(array,1,0)/moment(array,0,0)
             y = moment(array,0,1)/moment(array,0,0)
             return (x,y)

         def translate(array,x,y):
             buffer = np.roll(array,-x,axis=1)
             buffer = np.roll(buffer,-y,axis=0)
             return buffer

         def centeredarray(array):
             buffer = []
             for pic in tqdm(array):
                 shape = pic.shape
                 centroid = center(pic)
                 delta_x = -shape[1]/2 + centroid[0]
                 delta_y = -shape[0]/2 + centroid[1]

                 buffer += [translate(pic,int(delta_x),int(delta_y))]
             return buffer
```

```
In [39]: test = np.array(centeredarray(X[:640]))
```

```
HBox(children=(IntProgress(value=0, max=640), HTML(value='')))
```

```
In [5]: dataL = np.load("hirag.npz")['arr_0']
```

```
In [9]: data = np.array(dataL[:1599])
        y = np.array(Y[:1599])
```

```
In [14]: datax = data.reshape(data.shape[0], data.shape[1]*data.shape[2])
```

```
In [12]: from sklearn.cluster import KMeans
```

```
In [25]: kmeans = KMeans(n_clusters=10, random_state=0).fit(datax)
In [17]: from sklearn.metrics.cluster import adjusted_rand_score

In [26]: kmLabel = kmeans.labels_

In [27]: adjusted_rand_score(y,kmLabel)

Out[27]: 0.3869665701454988

In [28]: from sklearn import metrics

In [31]: metrics.silhouette_score(datax,kmLabel, metric='euclidean')

Out[31]: -0.016696665638428632

In [32]: metrics.homogeneity_score(y, kmLabel)

Out[32]: 0.5605506711832837

In [33]: metrics.adjusted_mutual_info_score(y, kmLabel)

/home/hexagramg/exp/venv/lib/python3.6/site-packages/sklearn/metrics/cluster/supe
  FutureWarning)


Out[33]: 0.5555767757562253

In [36]: from sklearn.cluster import AgglomerativeClustering

In [41]: agc = AgglomerativeClustering(n_clusters=10).fit(datax)

In [42]: agLabels = agc.labels_

In [43]: adjusted_rand_score(y,agLabels)

Out[43]: 0.47397686046328175

In [44]: metrics.silhouette_score(datax,agLabels, metric='euclidean')

Out[44]: -0.02925792626053541

In [45]: metrics.homogeneity_score(y, agLabels)

Out[45]: 0.6467588865151238

In [46]: metrics.adjusted_mutual_info_score(y, agLabels)

/home/hexagramg/exp/venv/lib/python3.6/site-packages/sklearn/metrics/cluster/supe
  FutureWarning)


Out[46]: 0.6427628965554346

In [47]: from sklearn.cluster import MeanShift

In [48]: clustering = MeanShift().fit(datax)
In [49]: mshLabels = clustering.labels_

In [50]: adjusted_rand_score(y,mshLabels)

Out[50]: 0.0020420433861530374

In [51]: metrics.silhouette_score(datax,mshLabels, metric='euclidean')

Out[51]: 0.0805389671008504

In [52]: metrics.homogeneity_score(y, mshLabels)

Out[52]: 0.10042434295351106

In [53]: metrics.adjusted_mutual_info_score(y, mshLabels)

/home/hexagramg/exp/venv/lib/python3.6/site-packages/sklearn/metrics/cluster/super
  FutureWarning)


Out[53]: 0.003904343710372022
```