Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа программной инженерии

Лабораторная работа №5

по дисциплине «Машинное обучение»

Выплолнил студент
гр. 33534/5 Стойкоски Н.С.

Руководитель И.А. Селин

Санкт-Петербург
2019 г.

Оглавление

Постановка задачи

1) Разбейте множество объектов из набора данных pluton.csv на 3 кластера методом центров тяжести (kmeans). Сравните качество разбиения в зависимости от максимального числа итераций алгоритма (визуально и по подходящим метрикам из sklearn.metrics).

2) Сгенерируйте набор данных в двумерном пространстве, состоящий из 3 кластеров, каждый из которых сильно “вытянут” вдоль одной из осей. Исследуйте качество кластеризации методом k-медоидов или clara (например, https://github.com/salspaugh/machine_learning/blob/master/clustering/kmedoids.py) в зависимости от 1) использования стандартизации (sklearn.preprocessing.StandardScaler); 2) типа метрики (достаточно евклидовой и манхэттенской). Объясните полученные результаты. Альтернативные реализации методов k-медоидов и clara допустимы.

3) Постройте дендрограмму для набора данных votes.csv (число голосов, поданных за республиканцев на выборах с 1856 по 1976 год). Строки представляют 50 штатов, а столбцы - годы выборов (31). Проинтерпретируйте полученный результат.

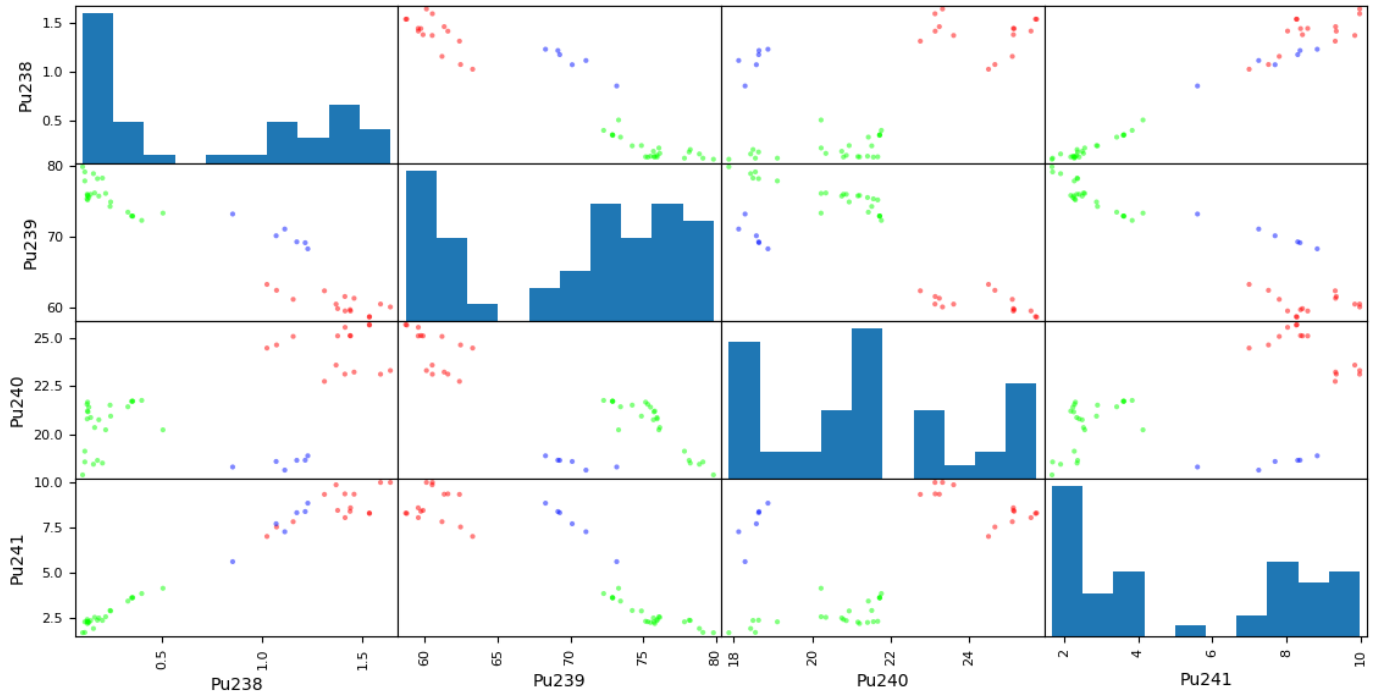
Ход работы

1. Была выполена кластеризация на наборе pluton.csv методом центров тяжести (kmeans), результаты кластеризации показаны в виде графиков (библиотека matplotlib), позволяющие визуализировать данных размерности большей, чем 2. Так же были вычислени оценки качества кластеризации (silhouette_score, calinski_harabaz_score, davies_bouldin_score). При этом были использовани разные значения максимального числа итерации при выполнения кластеризации, однако было найдено что на данном наборе достаточно всего 1 итерация для получения оптимального разбиения.

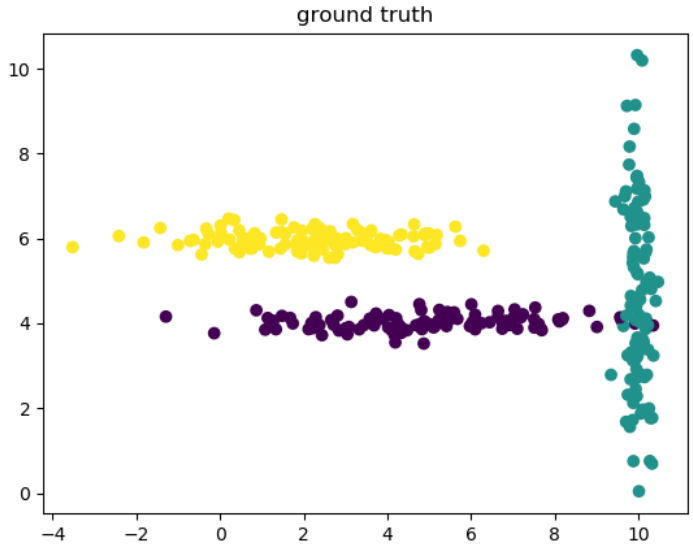
2. Был сгенерирован набор данных в двумерном пространстве, состоящий из 3 кластеров с использованием функции sklearn.datasets.make_blobs. Результирующие кластеры были вытянуты вдоль одной из осей при выполнения сдвига по одной из координат для каждой точки пропорционально расстоянию до центра кластера. Далее была выполнена кластериазция с использованием метода к-медоидов (https://github.com/salspaugh/machine_learning/blob/master/clustering/kmedoids.py), при этом были рассмотрены разные метрики расстояния (euclidean, manhattan), с использованием стандартизации (sklearn.preprocessing.StandardScaler) и без. Было найдено что, лучшие результаты получаются при использовании метрики ‘manhatan’ с использованием стандартизации.

3. Была построена дондрограмма для набора данных votes.csv. Результирующий график отображает близость числа голосов для разных штатах. По дендрограммы видно, что на выборах голосовалось за 2 основные партии, и хотя некоторые штаты технически проголосовали за республиканцев (или демократов), их население избирателей может быть весьма разнородным по своим предпочтениям.

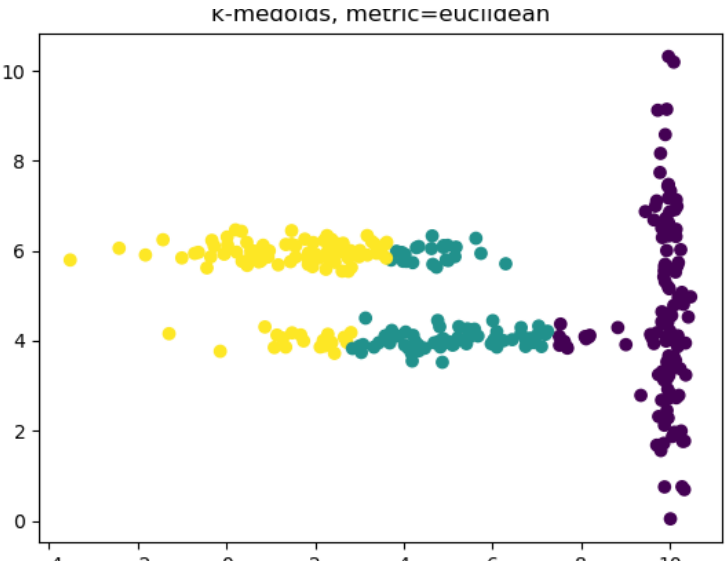
Результаты

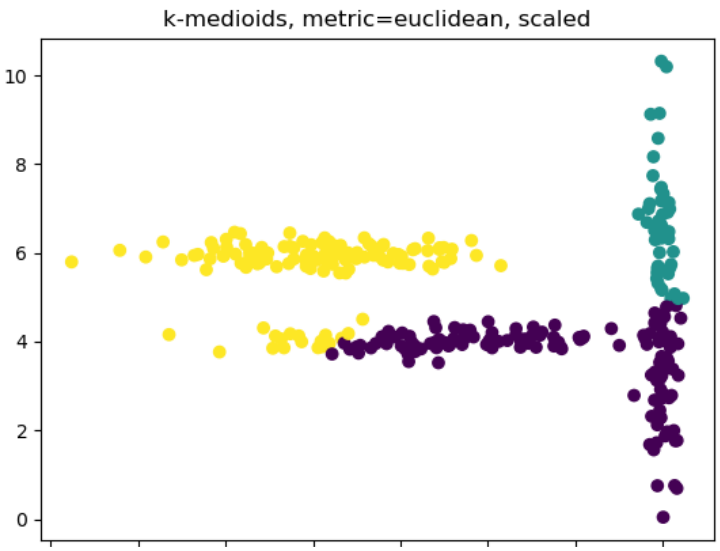


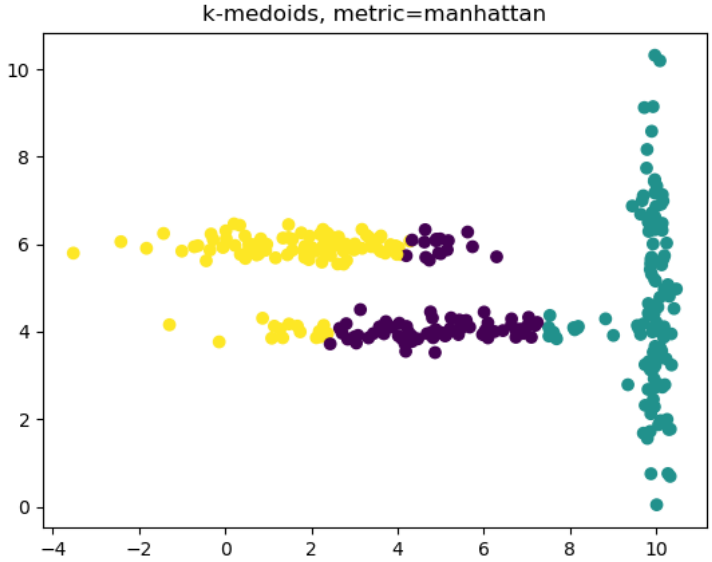

```
n_iter = 1
Silhouette-Score =  0.6678899065840955
Calinski-Harabaz Index =  243.31015661304917
Davies-Bouldin Index = 0.4183126928050032

n_iter = 2
Silhouette-Score =  0.6678899065840955
Calinski-Harabaz Index =  243.31015661304917
Davies-Bouldin Index = 0.4183126928050032
```

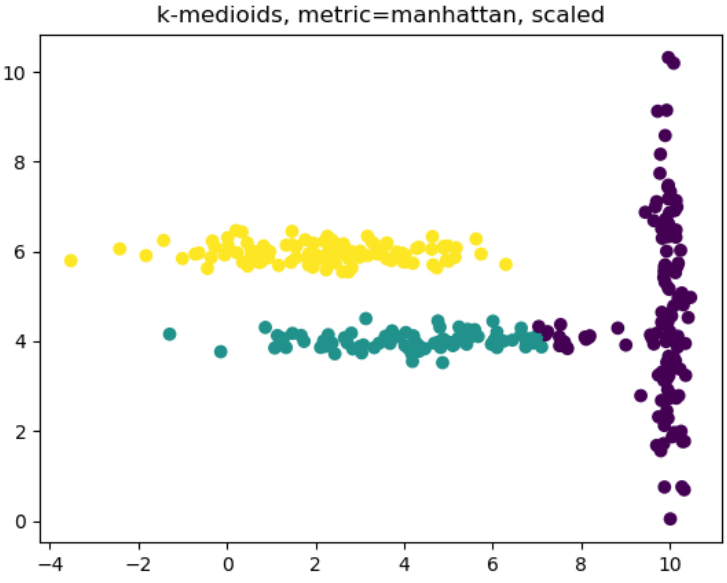


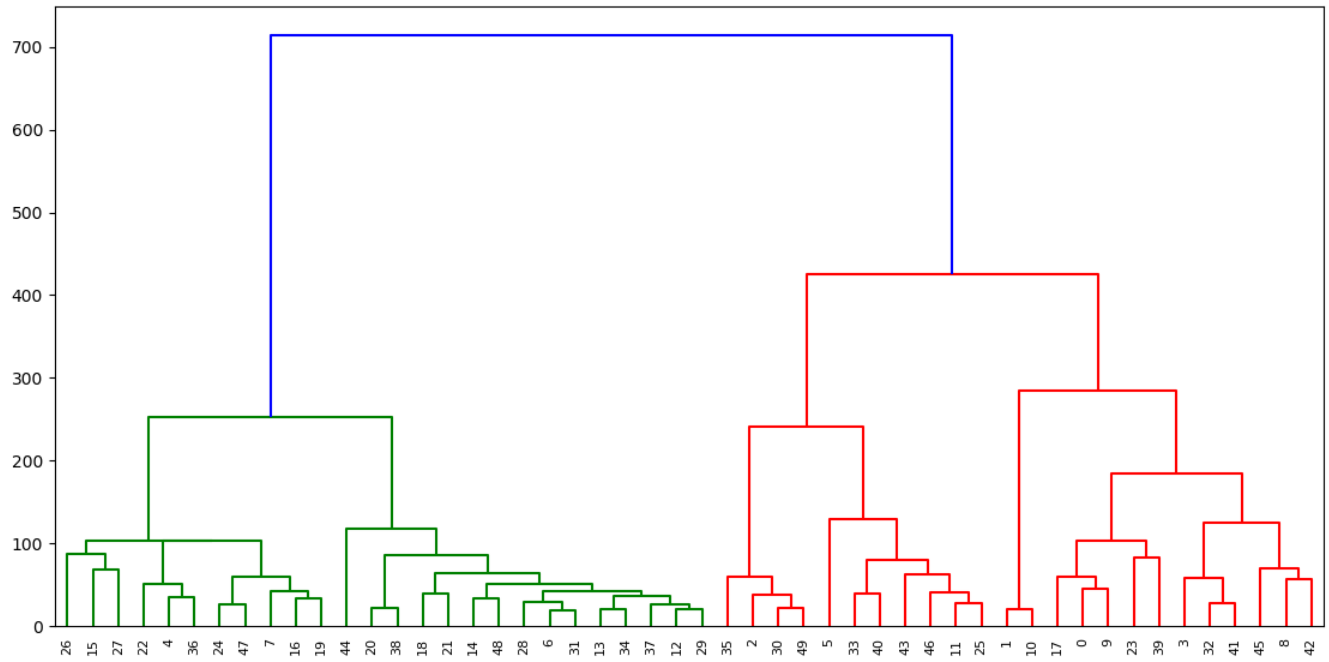

```
k-means, metric = euclidean
homogeneity-score = 0.560
completeness-score = 0.564

k-means, metric = euclidean, scaled
homogeneity-score = 0.584
completeness-score = 0.642

k-means, metric = manhattan
homogeneity-score = 0.606
completeness-score = 0.610

k-means, metric = manhattan, scaled
homogeneity-score = 0.841
completeness-score = 0.850
```











Вывод

В ходе данной лабораторной работы был получен опыт работы с методами k-средных, k-медоидов для задач кластеризации. Было исследовано как разные метрики расстояния и стандартизация данных влияет на качество кластеризации. Так же был получен опыт работы с дендрограммы, как и построения графиков для визуализация данных размерности, большей чем 2.

Текст программы

```
from sklearn.cluster import KMeans
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.colors
import matplotlib.cm
import pandas.plotting
from sklearn import metrics
from sklearn.datasets import make_blobs
from sklearn.preprocessing import StandardScaler

def task1():
    X = pd.read_csv("pluton.csv")

    for numIterations in [1, 2]:

        kmeans = KMeans(n_clusters=3, random_state=0, max_iter=numIterations).fit(X)

        colormap = plt.get_cmap('hsv')
        norm = matplotlib.colors.Normalize(vmin=0, vmax=3)
        axes = pd.plotting.scatter_matrix(X, color=colormap(norm(kmeans.labels_)))
        plt.suptitle(f'max_iter = {numIterations}')

        labels = kmeans.labels_
        print(f'max_iter = {numIterations}, n_iter = {kmeans.n_iter_}')
        print('Silhouette-Score = ', metrics.silhouette_score(X, labels, metric='euclidean'))
        print('Calinski-Harabaz Index = ', metrics.calinski_harabaz_score(X, labels))
        print('Davies-Bouldin Index', metrics.davies_bouldin_score(X, labels), end='\n\n')

    matplotlib.pyplot.show()


def task2():

    centers = [[5, 4], [10, 4.5], [2, 6]]
    X, y = make_blobs(n_samples=300, n_features=2, centers=centers, cluster_std=0.2,
random_state=5)

    for i in range(X.shape[0]):
        distance_to_center = X[i][y[i]%2] - centers[y[i]][y[i]%2]
        X[i][y[i]%2] += 10 * distance_to_center

    plt.figure()
    plt.scatter(X[:, 0], X[:, 1], c=y)
    plt.title("ground truth")

    scaler = StandardScaler(with_mean=True, with_std=True)
    X_scaled = scaler.fit_transform(X)

    from ml5_ import kmedoids
    from sklearn.metrics.pairwise import pairwise_distances
    from sklearn.metrics import homogeneity_score
    from sklearn.metrics import completeness_score

    for metric in ['euclidean', 'manhattan']:

        D = pairwise_distances(X, metric=metric)
```

```
        clusters, medoids = kmedoids.cluster(D, 3)

        color_dict = {medoids[0]: 0, medoids[1]: 1, medoids[2]: 2}
        clusters = [color_dict[y] for y in clusters]
        print(f'k-means, metric = {metric}')
        print(f'homogeneity-score = {homogeneity_score(y, clusters)}')
        print(f'completeness-score = {completeness_score(y, clusters)}')
        plt.figure()
        plt.scatter(X[:, 0], X[:, 1], c=clusters, marker='o')
        plt.title(f'k-medoids, metric={metric}')


        D_scaled = pairwise_distances(X_scaled, metric=metric)
        clusters, medoids = kmedoids.cluster(D_scaled, 3)

        color_dict = {medoids[0]: 0, medoids[1]: 1, medoids[2]: 2}
        clusters = [color_dict[y] for y in clusters]
        print(f'k-means, metric = {metric}, scaled')
        print(f'homogeneity-score = {homogeneity_score(y, clusters)}')
        print(f'completeness-score = {completeness_score(y, clusters)}')
        plt.figure()
        plt.scatter(X[:, 0], X[:, 1], c=clusters, marker='o')
        plt.title(f'k-medioids, metric={metric}, scaled')

def task3():
    df = pd.read_csv('votes.csv')

    df = df.fillna(0)
    print(df)
    from scipy.cluster.hierarchy import linkage, fcluster, dendrogram

    Z = linkage(df, method='ward')

    plt.figure()
    dendrogram(Z)
    plt.show()


task1()
task2()
task3()
plt.show()
```