

Lab 7 实验题目

学号	陈政培	学院	专业
姓名	17363011	智能工程学院	智能科学与技术

1、实验目的：k-means 应用实践

2、实验环境：vs code、python3.6.8、百度 Web API

3、实验步骤：

① 获取地图数据

在百度地图开放平台中创建应用时填写访问方的公网 IP，一开始在 cmd 中输入 ipconfig /all 获取本机 ip 地址

```
以太网适配器 以太网:

    连接特定的 DNS 后缀 . . . . . : sysu.edu.cn
    描述. . . . . : Realtek PCIe GBE Family Controller
    物理地址. . . . . : 78-24-AF-43-42-E2
    DHCP 已启用 . . . . . : 是
    自动配置已启用. . . . . : 是
    IPv6 地址. . . . . : 2001:250:3002:44b0:9cda:c29f:9705:d706(首选)
    临时 IPv6 地址. . . . . : 2001:250:3002:44b0:555:8057:1501:2df2(首选)
    本地链接 IPv6 地址. . . . . : fe80::9cda:c29f:9705:d706%9(首选)
    IPv4 地址. . . . . : 172.18.97.189(首选)
    子网掩码 . . . . . : 255.255.254.0
    获得租约的时间 . . . . . : 2019年10月30日 15:37:27
    租约过期的时间 . . . . . : 2019年10月30日 21:37:29
    默认网关. . . . . : fe80::3a22:d6ff:fee6:7a0%9
                       172.18.97.254
    DHCP 服务器 . . . . . : 222.200.160.1
    DHCPv6 IAID . . . . . : 74982575
    DHCPv6 客户端 DUID . . . . . : 00-01-00-01-25-3B-6D-A7-78-24-AF-43-42-E2
    DNS 服务器 . . . . . : 10.8.8.8
                       10.8.4.4
    TCP/IP 上的 NetBIOS . . . . . : 已启用
```

误以为在 IP 白名单处填写对应的网关 IP 就可以，结果出现报错

```
{'status': 210, 'message': 'APP IP校验失败'}
Traceback (most recent call last):
  File "c:/Users/93744/Desktop/code_and_data/api.py", line 37, in <module>
    geoGrab()
  File "c:/Users/93744/Desktop/code_and_data/api.py", line 30, in geoGrab
    lat = hjson['results'][i]['location']['lat']
KeyError: 'results'
```

发现 APP IP 校验失败，后来发现原来此处 cmd 获取的 ip 仍然是对于学校内网的 ip。为了获取公网 ip，我打开了 <http://txt.go.sohu.com/ip/soip> 以检查我的实际访问公网 ip

```
String.prototype.getQueryString=function(v){var reg=new RegExp("(^|&|\\?)"+v+"(=|&|\\?)(&|$)"); r;if(r=this.match(reg)){return unescape(r[2]);}return null;}var sohu_IP_Loc="sohu.com";LocUrl=document.location.href;if((LocUrl.indexOf("sohusce.com")>=0)|| (LocUrl.indexOf("sohu.com")>=0)|| (LocUrl.indexOf("chinaren.com")>=0)|| (LocUrl.indexOf("17173.com")>=0)|| (LocUrl.indexOf("focus.cn")>=0)){window.sohu_user_ip="120.236.174.170";sohu_IP_Loc="CN990000";sohu_IP_Loc_V="CN";}var AdLoc2=sohu_IP_Loc.substr(0,2),AdLoc4=sohu_IP_Loc.substr(0,4),AdLoc6=sohu_IP_Loc.substr(0,6);if(window.location.href.getQueryString("ip"))sohu_IP_Loc=AdLoc2=AdLoc4=AdLoc6=window.location.href.getQueryString("ip");
```

原来我的实际公网 ip 是中山大学的 120.236.174.170

- ② 将应用 AK 放到 python 代码中，成功获取了 Restaurant_Data_Beijing.txt
- ③ 测试 loadDataSet()、randCent() 和 distEclud() 函数
- ④ kMeans 算法实现

在未修改 kmeans.py 文件前，可以直接显示测试结果，但是当把语句“from numpy import *”修改成“import numpy as np”后代码出现了报错

```
Traceback (most recent call last):
  File "c:/Users/93744/Desktop/code_and_data/main.py", line 14, in <module>
    myCentroids, clustAssing = kmeans.kMeans(dataset, 4)
  File "c:/Users/93744/Desktop/code_and_data/kmeans.py", line 86, in kMeans
    if distJI < minDist:
ValueError: The truth value of an array with more than one element is ambiguous.
Use a.any() or a.all()
```

本以为是语句

```
if distJI < minDist:
```

是数据结构之间的比较出现了问题，后来才发现 python 中除了 numpy 库中有一个 sum 函数，python 本身也有一个 sum 函数。但是两者实现的功能和输出结果并不相同，所以出现了数据结构的错误

```
return np.sqrt(np.sum(np.power(vecA - vecB, 2)))
```

需要将 kmeans.py 文件 19 行的代码中 sum 改成 np.sum 即可正常运行

- ⑤ 用 kMeans 算法对地图上的点聚类

原理和之前使用 testSet 一样

4、实验结果与分析：

- ① 从文本文件构建矩阵并建立辅助函数

```
[[ -5.379713]]
[[ -4.232586]]
[[ 5.1904]]
[[ 4.838138]]
[[ 3.5067209 -0.69521633]
 [ 0.29318168 3.87091094]]
5.184632816681332
```

输出结果分别为

```
print(min(dataset[:,0]))
print(min(dataset[:,1]))
print(max(dataset[:,1]))
print(max(dataset[:,0]))
print(kmeans.randCent(dataset, 2))
print(kmeans.distEclud(dataset[0], dataset[1]))
```

② kMeans 算法实现

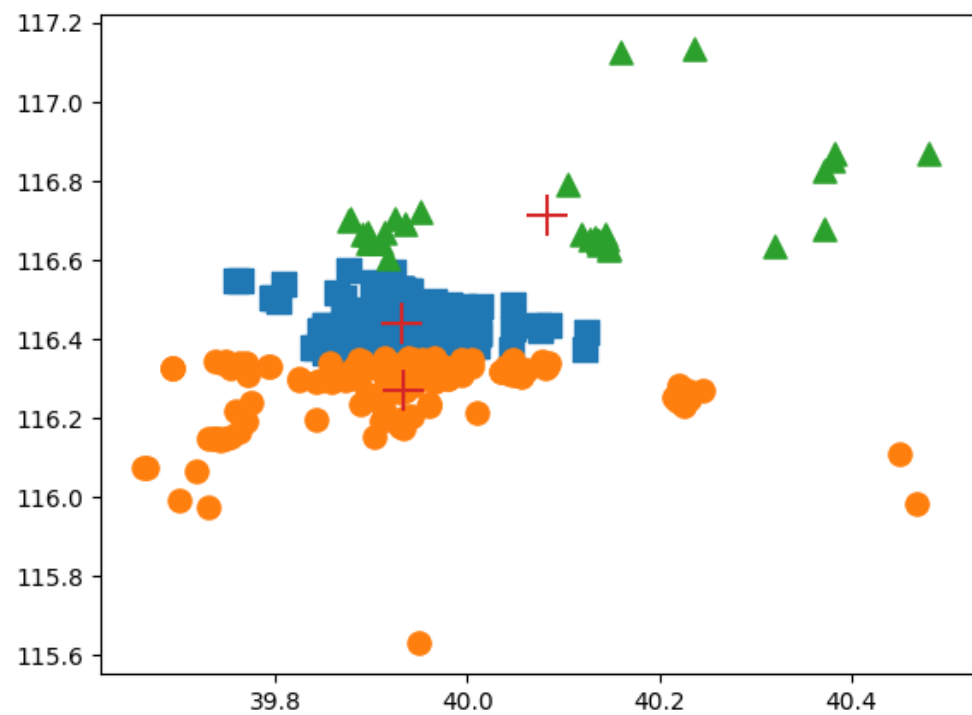
```
[[ -3.53973889 -2.89384326]
 [ -2.46154315  2.78737555]
 [  2.65077367 -2.79019029]
 [  2.6265299   3.10868015]]

[[ 3.      2.3201915 ]
 [ 1.      1.39004893]
 [ 2.      7.46974076]
 [ 0.      3.60477283]
 [ 3.      2.7696782  ]
 [ 1.      2.80101213]
 [ 2.      5.10287596]
 [ 0.      1.37029303]
 [ 3.      2.29348924]
 [ 1.      0.64596748]
 [ 2.      1.72819697]
 [ 0.      0.60909593]
 [ 3.      2.51695402]
 [ 1.      0.13871642]
 [ 2.      9.12853034]
 [ 2.     10.63785781]
 [ 3.      2.39726914]
 [ 1.      3.1024236  ]
 [ 2.      0.40704464]
 [ 0.      0.49023594]
 [ 3.      0.13870613]
 [ 1.      0.510241  ]
 [ 2.      0.9939764  ]
```

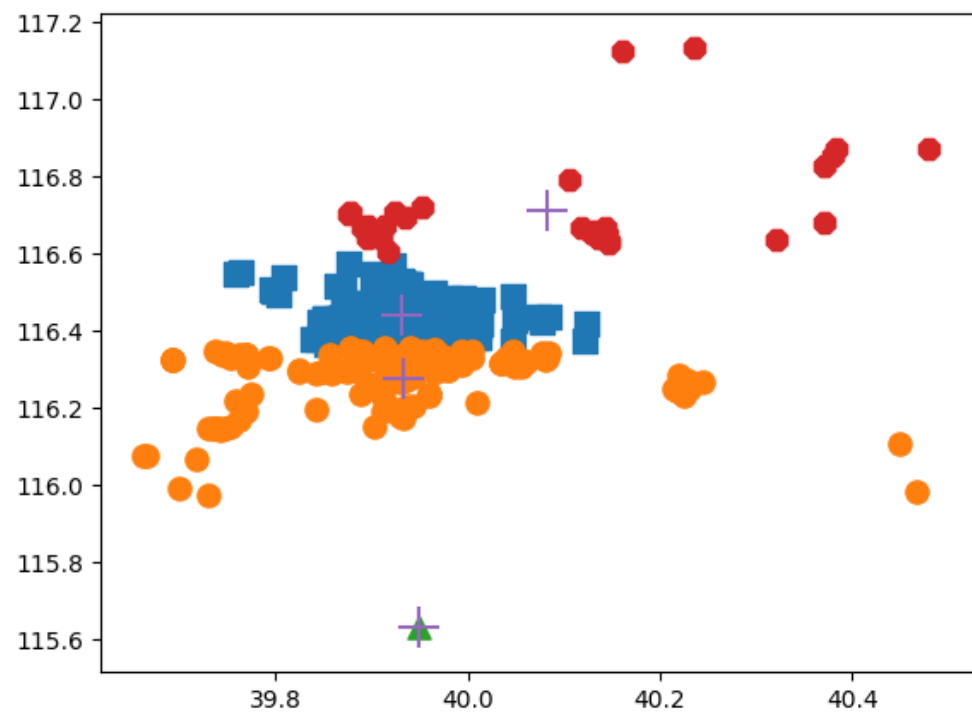
得到了 4 个质心的坐标和各个簇的分配结果（结果截图了一小部分）

③ 用 kMeans 算法对地图上的点聚类

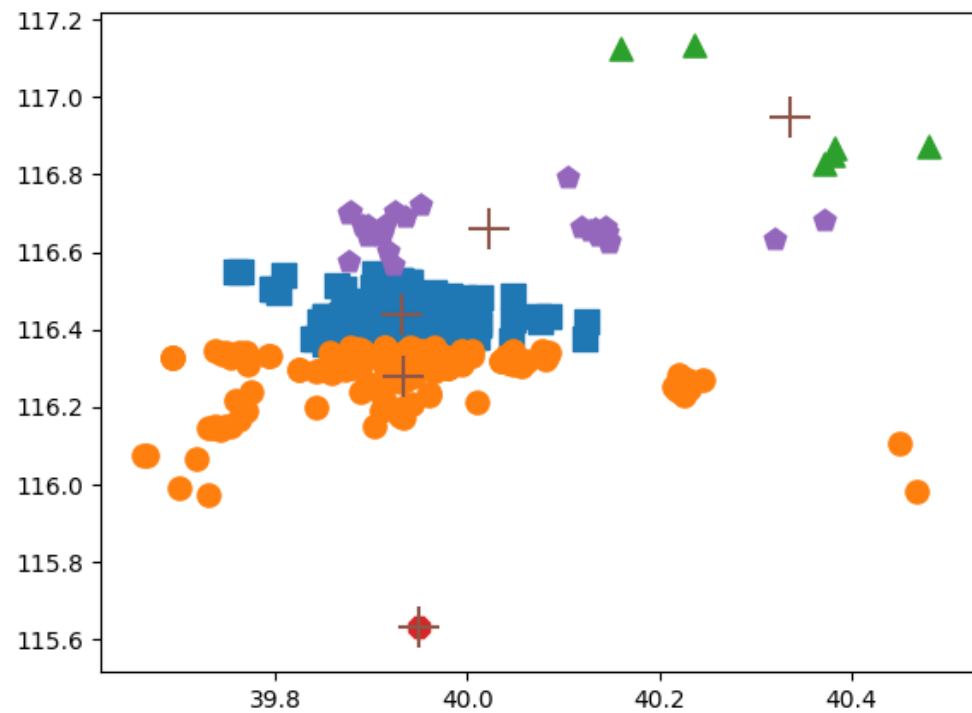
Kmeans.clusterPlaces(3)



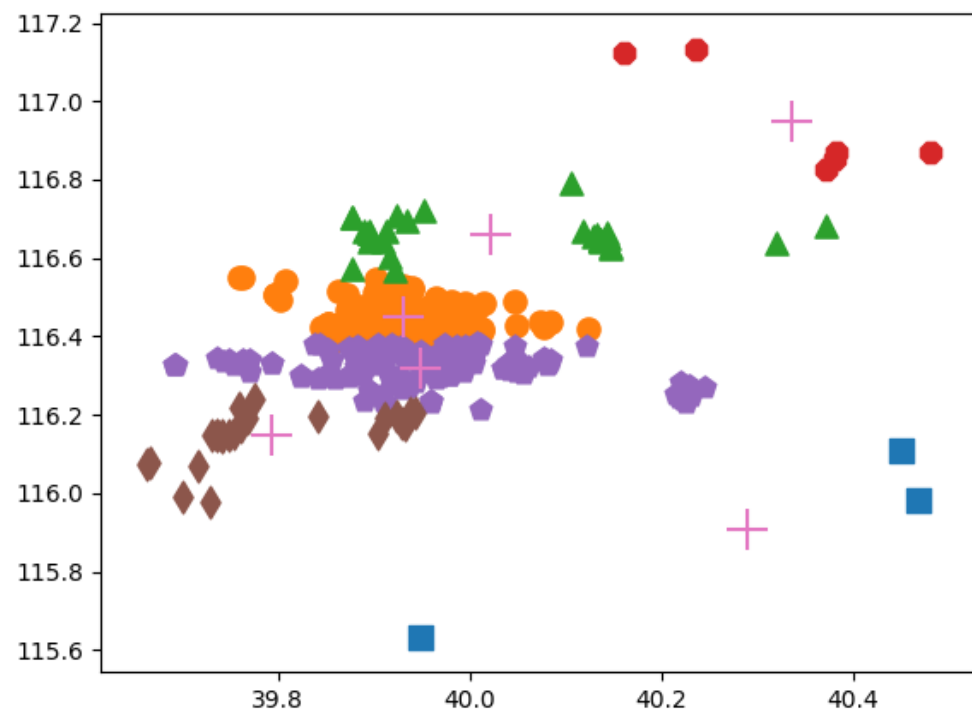
Kmeans.clusterPlaces(4)



Kmeans.clusterPlaces(5)



Kmeans.clusterPlaces(6)



5、作业：

① 对聚类结果可视化（包含样本点和簇中心，用不同颜色、记号标记）

4 结果展示中已经实现了

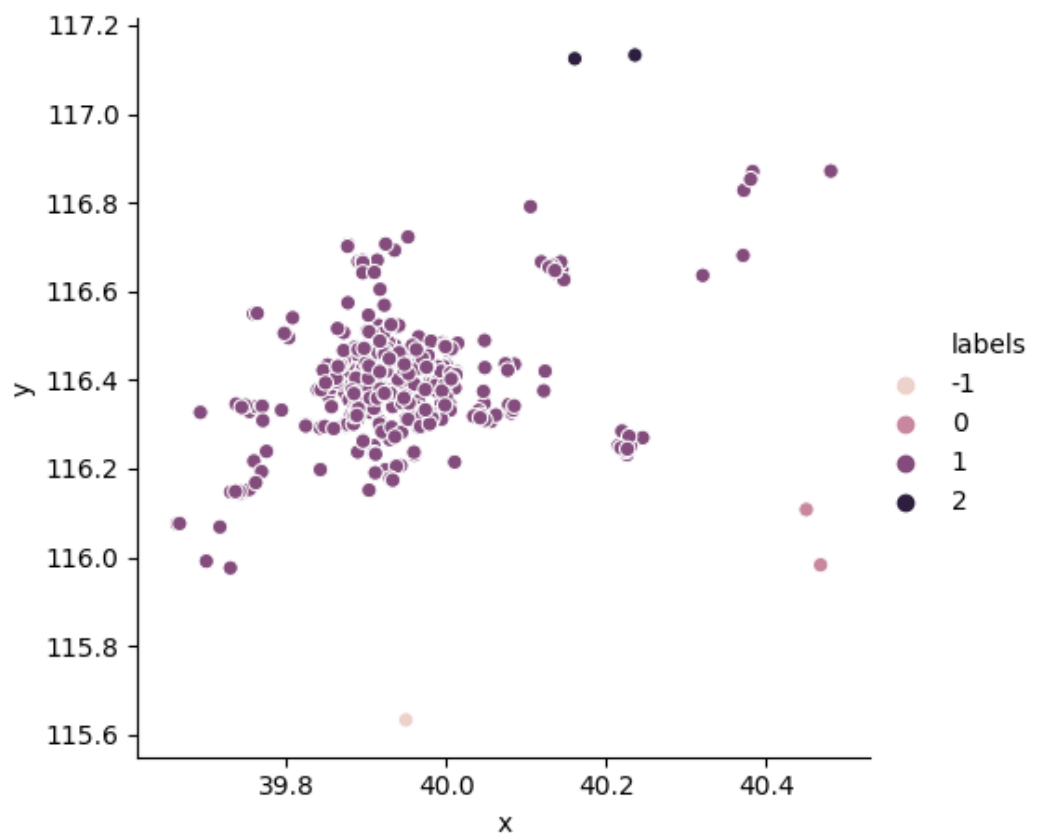
② Kmeans.py 中的语句代替并修改代码使其正常执行

修改结果见代码，遇到的问题和 bug 修正过程详见 3.④中描述

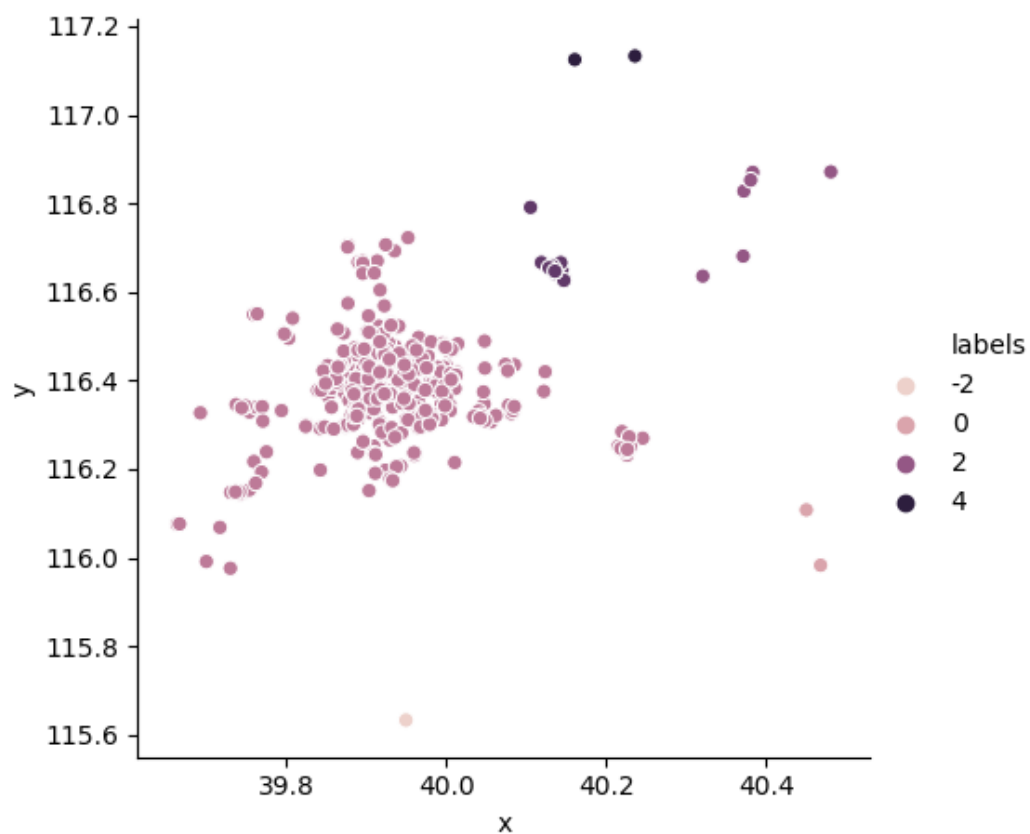
③ 尝试用 DBSCAN 聚类该数据集

使用了 sklearn 库中的 cluster 里的函数 KMeans 和 DBSCAN 进行聚类

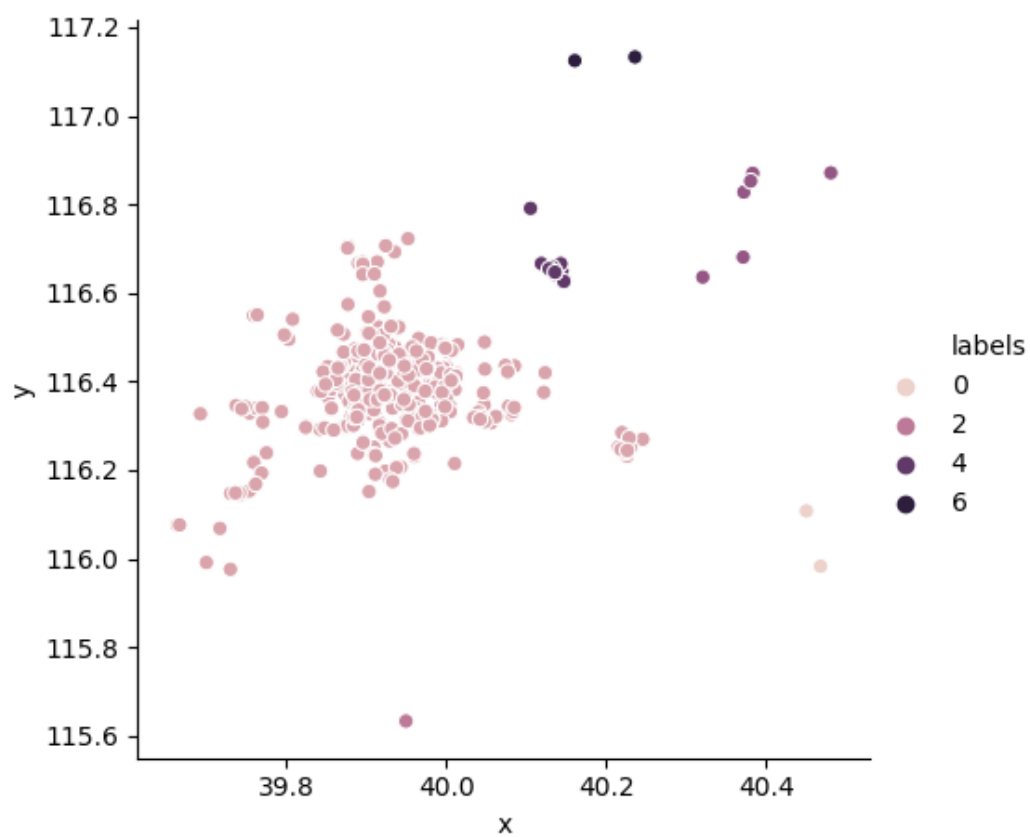
- 因为 DBSCAN 聚类函数是基于密度的聚类函数，所以不需要预先指定簇的个数，最终的簇的个数不确定。分类的依据是聚类的半径 ϵ 和邻域内最小点数目 min_samples
- 想要获得和 KMeans 相似的聚类效果则需要调试 ϵ 和 min_samples 的大小，以得到合适的簇数目
- $\epsilon = 0.2$, $\text{min_samples} = 2$ 时获得 3 个簇



- $\epsilon = 0.15$, $\text{min_samples} = 1.5$ 时获 5 个簇



- Eps = 0.15 , min_samples = 1 时获得 6 个簇



④ 利用轮廓系数评估 kMeans 和 DBSCAN 对该数据集的聚类效果

通过 sklearn 中 metrics 里的函数 silhouette_score 来对两种聚类算法的结果进行轮廓系数评估

- DBSCAN 函数在实际计算时把一部分餐厅作为噪声屏蔽了

- 簇为 3 时

```
KMeans:
噪声比: 0.00%
分簇的数目: 3
轮廓系数: 0.863
DBSCAN:
噪声比: 0.25%
分簇的数目: 3
轮廓系数: 0.833
```

- 簇为 5 时

```
KMeans:
噪声比: 0.00%
分簇的数目: 5
轮廓系数: 0.893
DBSCAN:
噪声比: 0.25%
分簇的数目: 5
轮廓系数: 0.855
```

- 簇为 6 时

```
KMeans:
噪声比: 0.00%
分簇的数目: 6
轮廓系数: 0.904
DBSCAN:
噪声比: 0.00%
分簇的数目: 6
轮廓系数: 0.858
```

- 整体结论为在轮廓系数上 DBSCAN 相对更优，但是有一定噪声损失

⑤ 尝试在 Tensorflow 环境下实现 kMeans

详见代码

```
the first cluster division is : [1 1 2 1 1 1 1 1 1 1 0 2 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 2 1
2 2 2 2 2 1 2 1 2 1 2 2 2 2 1 2 0 1 1 1 1 1 2 2 2 0 2 1 2 1 1 1 1 1 2 1 1
1 1 2 1 2 1 1 1 2 2 0 2 1 1 1 1 1 1 1 1 2 1 2 1 1 1 2 1 2 0 1 2 2 1 1 1
2 2 1 2 2 2 1 1 2 2 2 2 1 1 2 1 2 1 0 1 1 2 2 2 2 2 1 2 1 2 1 1 1 2 2 2
1 2 2 0 1 2 1 2 0 0 2 1 1 1 1 2 1 0 1 2 1 1 1 1 1 1 1 2 1 1 1 1 2 0 2
2 1 1 2 1 2 0 1 2 1 2 0 2 2 2 2 1 1 1 2 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1
1 2 2 1 1 0 1 1 1 2 1 2 1 2 2 1 1 2 2 1 1 0 1 2 1 1 0 1 1 1 1 1 1 1 1 2
2 2 1 1 1 2 1 0 1 2 1 2 1 2 1 2 1 2 1 2 1 1 2 1 1 2 1 1 1 1 1 1 1 1 2 1
1 1 2 1 2 2 2 1 1 1 1 2 1 0 1 1 0 1 1 1 2 2 2 1 2 1 2 1 1 1 1 1 2 1 1 1
1 1 1 2 1 0 2 1 2 2 1 1 1 1 2 1 1 2 2 1 1 2 1 0 1 1 1 1 2 1 0 2 1 0 1 2 2
1 2 1 0 1 2 1 1 2 1 1 1 1 2 2 1 0 1 1 2 1 1 1 1 2 1 1 1 1 0]
```

可以实现聚类结果，但是由于时间原因可视化未进行颜色分类

6、实验总结：

- APP IP 校验失败，后来发现原来此处 cmd 获取的 ip 仍然是对于学校内网的 ip。为了获取公网 ip，我打开了 <http://txt.go.sohu.com/ip/soip> 以检查我的实际访问公网 ip
- python 中除了 numpy 库中有一个 sum 函数，python 本身也有一个 sum 函数。但是两者实现的功能和输出结果并不相同，所以出现了数据结构的错误
- 调试 DBSCAN 代码时报错

```
Traceback (most recent call last):
  File "c:/Users/93744/Desktop/code_and_data/dbscan.py", line 43, in <module>
    sc_score = metrics.silhouette_score(km, labels, metric='euclidean')
  File "D:\Python36\lib\site-packages\sklearn\metrics\cluster\unsupervised.py", line 117, in silhouette_score
  File "D:\Python36\lib\site-packages\sklearn\metrics\cluster\unsupervised.py", line 212, in silhouette_samples
    X, labels = check_X_y(X, labels, accept_sparse=['csc', 'csr'])
  File "D:\Python36\lib\site-packages\sklearn\utils\validation.py", line 719, in check_X_y
    estimator=estimator)
  File "D:\Python36\lib\site-packages\sklearn\utils\validation.py", line 514, in check_array
    "if it contains a single sample.".format(array))
ValueError: Expected 2D array, got scalar array instead:
array=KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
              n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
              random_state=None, tol=0.0001, verbose=0).
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
```

原因是 silhouette_score 函数接受的参数中需要的 list 类型是 (1, -1) 的，而我们传进去的数据 shape 是错误的，对相应数据进行 reshape 即可

- 在网上找到的教程使用的 tensorflow 版本是 1.0，所以很多 tf 函数都不能直接使用，必须通过 tf.compat.v1. 来访问老版本的函数

7、参考文献：

- ① <https://www.jianshu.com/p/82db047eea13>
- ② <https://blog.csdn.net/u012967763/article/details/79149703>
- ③ https://github.com/aliceleee/KMeans_Tensorflow/tree/master/code