

## Lab7 k-Means 应用实践

**实验简介:** 利用 Python 实现 k-Means 聚类。使用 Python 访问 Baidu Web 的 API, 先用 Baidu Web 的 API 获得数据, 然后用 kMeans 算法对地理位置进行聚类, 并对聚类得到的簇进行后处理。代码文件: kmeans.py。

**作业要求:** 作业要求: 见 QQ 群文件: “作业要求.pdf”

### 1、获取地图数据

百度地图提供 API 的网址:

<http://lbsyun.baidu.com/index.php?title=webapi/guide/webservice-placeapi>

#### (1) 注册成为开发者

使用限制:

Place API 是一套免费使用的API接口, 调用次数限制为10万次/天。

ak是API请求串的必填参数, 请先[获取密钥](#), 若无百度账号则首先需要[注册百度账号](#)。

同一个帐号下的HTTP/HTTPS请求, 配额、并发共享。

注册登录百度账号之后, 点击获取密钥, 弹出注册成为开发者页面:



点击创建应用:



**注意:** IP 白名单中要填入访问方的公网 IP, 查看 IP 白名单中的 IP 是否与本机 IP 一致, 若不一致则改成本机 IP, 这里是中大公网 IP。为了获取 IP, 打开 <http://txt.go.sohu.com/ip/soip> 以检查本机实际访问公网的 IP。

应用类型选择服务端，可以看到下面有需要的 Place API v2 服务。点击确定就可以看到秘钥，如下（AK 部分）：

创建应用

回收站

每页显示30条

应用编号	应用名称	访问应用（AK）	应用类别	备注信息 （双击更改）	应用配置
9353640	example		服务端		设置 删除

### （2）使用 API 获取数据（api.py）

查看网页，了解 API 的使用格式：

http://api.map.baidu.com/place/v2/search?q=查询内容&page\_size=范围记录数量  
&page\_num=分页页码&region=地区&output=数据格式&ak=秘钥

# 打开命令窗口，并切换至文件“api.py”所在目录，运行：

```
> python api.py
```

观察获得的数据和目录下的 Restaurant\_Data\_Beijing.txt 文件中保存的数据是一致的：

```
C:\Users\ZGL\Desktop>python api.py
40.680862 117.210130
40.665416 117.240121
40.691511 117.177271
40.686906 117.187322
40.690545 117.181281
40.685128 117.164653
40.691104 117.178515
40.690412 117.194770
40.608046 117.123885
40.645886 117.134904
40.546001 117.129393
40.54014 117.117611
40.512642 117.080514
40.745929 116.896916
```

## 2、实现 kMean 聚类

### （1）建立辅助函数

函数 loadDataSet() 读取文件的数据，函数 distEclud() 计算两个向量的欧式距离，函数 randCent() 随机生成 k 个随机质心。

```
>>> import kmeans
# 从文本文件中构建矩阵：
>>> dataset = mat(kmeans.loadDataSet('testSet.txt'))
# 测试函数 randCent():
>>> min(dataset[:, 0])
>>> min(dataset[:, 1])
>>> max(dataset[:, 1])
>>> max(dataset[:, 0])
# 查看 randCent() 能否生成 min 到 max 之间的值：
>>> kmeans.randCent(dataset, 2)
```

# 测试距离计算函数:

```
>>> kmeans.distEclud(dataset[0], dataset[1])
```

### (2) kMean 算法实现

函数 `kMeans()` 接受 4 个输入参数, 数据集及簇的数目是必选参数, 而计算距离和创建初始质心的函数可选。

```
>>> dataset = mat(kmeans.loadDataSet('testSet.txt'))
```

# 查看聚类结果:

```
>>> myCentroids, clustAssing = kmeans.kMeans(dataset, 4)
```

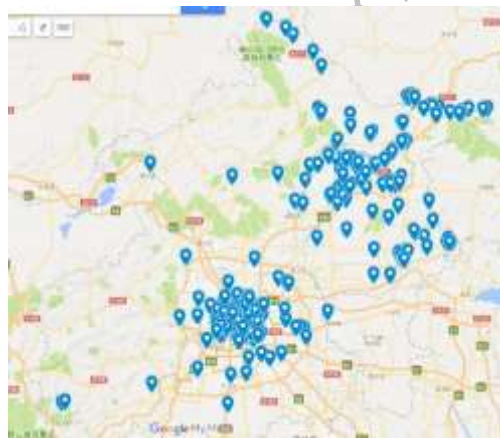
# 参看迭代的次数及结果:

```
>>> myCentroids
```

```
>>> clustAssing
```

### 3、利用 kMeans 对地图上的点聚类

餐厅是一个城市的重要组成部分, 在北京城内有不少餐厅, 当今地政府想要建立 4 个餐厅管理服务点, 对整个北京中的餐厅进行管理, 但是无法确定管理服务点要建在哪里才比较合理? 假设现在给出北京地区的一些饭店所在的经纬度, 具体分布如下图所示。尝试利用 `kMeans` 依据饭店的分布, 找其各部分的中心位置。



#### (1) 准备数据

饭店的经纬度数据存放在 `Restaurant_Data_Beijing.txt` 文件中, 其中每一行数据的第一列代表地点的纬度 (北纬), 第二列代表经度 (东经):

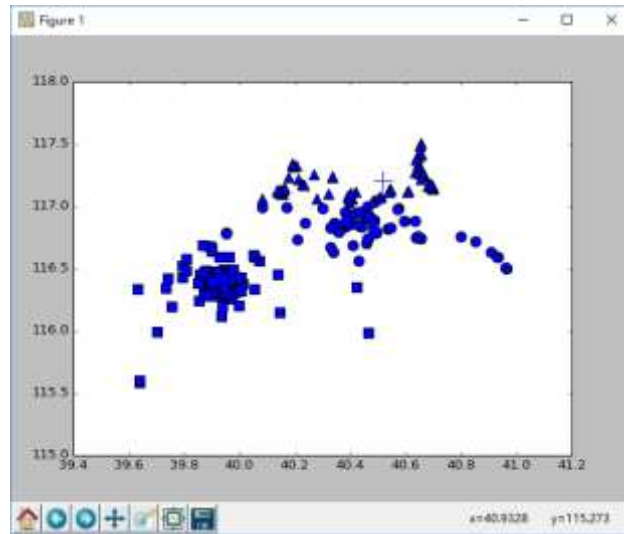
```
>>> dataMat = kmeans.loadDataSet('Restaurant_Data_Beijing.txt')
```

```
>>> dataMat
```

#### (2) 对地理坐标进行聚类

函数 `distSLC()` 返回地球表面两点之间的距离, 函数 `clusterPlaces()` 将文本文件中的地点进行聚类并画出结果。

```
>>> kmeans.clusterPlaces(3)
```



可与 google map 中的标记进行对比。

不同簇的数据点用不同的形状标记，+号所标注的就是对应簇的质心。可看到地点被大致分成 3 部分。依次修改 k 值为 4、5、6，观察相应的图像输出：

```
>>> kmeans.clusterPlaces(4)
```

```
>>> kmeans.clusterPlaces(5)
```

```
>>> kmeans.clusterPlaces(6)
```

#### 4、操作习题

(1) 利用 `sklearn.cluster.kMeans` 聚类该数据集，并访问一些重要属性和方法，用轮廓系数评估不同 k 值的聚类效果，选择最佳 k 值。

(2) 利用 `sklearn.mixture.GaussianMixture` 聚类该数据集，并访问一些重要属性和方法，测试不同的参数 `covariance_type` 值的效果。

(3) 实现 PPT 中的示例 1、示例 2