****

**机器学习**

**实验报告**

**学 号 U201913602**

**姓 名 莫茗程**

**专 业 电信1904**

**任 课 教 师 王邦**

**院（系、所） 电信学院**

**2022年 5月19日**

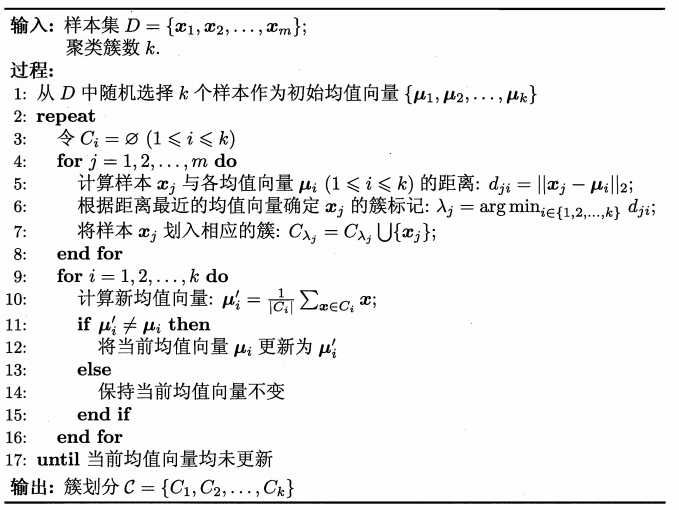
**任务1：对地理数据应用二分k-均值算法聚类**

1.任务要求：

你的朋友Drew希望你带他去城里庆祝他的生日。由于其他一些朋友也会过来，所以需要你提供一个大家都可行的计划。Drew给了你希望去的69个地址和相应的经纬度。你要决定将这些地方进行聚类的最佳策略，这样可以安排交通工具抵达这些簇的质心，然后步行到每个簇内地址。

2.实验原理：

2.1 k-means算法如下：



k-means算法是直接随机选取k个样本作为初始均值向量，然后再进行更新。

k-means算法经典简单快速，但是对于某些场景下不够适用。对初值敏感，选取的初始均值向量不同，则可能导致不同结果。

k-means算法不适合发现非凸形状的簇或者大小差别很大的簇，且对于“噪声”和孤立点数据敏感。

2.2 二分k-means算法：

为克服K-means算法收敛于局部最小值的问题，因此有人提出二分k-means算法，该算法首先将所有点作为一个簇，然后将该簇一分为二。之后选择其中一个簇继续进行划分，选择哪一个簇进行划分取决于对其划分是否可以最大程度降低误差平方和(SSE)的值。上述基于SSE的划分过程不断重复，直到得到用户指定的簇数目为止。

2.3 二分k-means算法伪代码：

将所有点看成一个簇

当簇数目小于k时

对于每一个簇

计算总误差

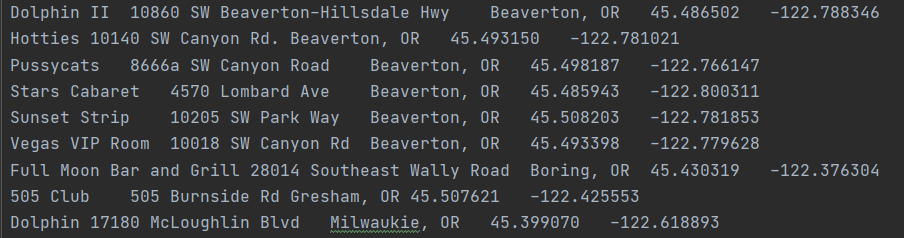
在给定的簇上面进行k-均值聚类（k=2）

计算将该簇一分为二之后的总误差

选择使得误差最小的那个簇进行划分操作

2.4 数据集分析

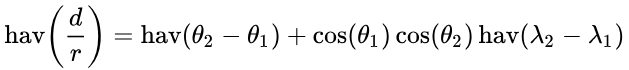
数据集在places.txt中，一共69条数据，每条数据的最后两列为该地点的经纬度，即在地图上的坐标。部分数据集如下：



2.5 距离计算

Demo中采用计算两点的球面距离。已知两地的经纬度，计算球面距离，使用Haversine(半正矢公式)

根据Haversine公式有



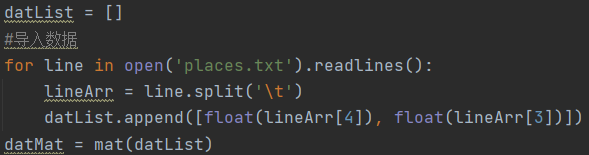
其中分别为纬度和经度。d为两点的球面距离，r为地球半径

最终算得

3.实验思路：

3.1 读取数据

读取每一行文本，用split(‘\t’)进行分割，并保存我们需要的经纬度信息



3.2 二分k-means算法

按照伪代码所写，先将所有样本点看作一个簇，簇的中心为样本位置的平均

centroid0 = mean(dataSet, axis=0).tolist()[0]

然后把该点加入簇中心列表

centList =[centroid0]

循环遍历计算每个样本点到簇中心的误差。

clusterAssment[j,1] = distMeas(mat(centroid0), dataSet[j,:])\*\*2

算得每个样本到簇中心的误差后，开始循环划分新簇，每次划分采用的是k-means(k=2)算法。

我们要找出使得划分后SSE最小的簇进行划分，所以循环每一个centList划分一次找到需要继续划分的簇。划分的方法为:

centroidMat, splitClustAss = kMeans(ptsInCurrCluster, 2, distMeas)

划分之后计算二分前后的SSE

sseSplit = sum(splitClustAss[:,1])

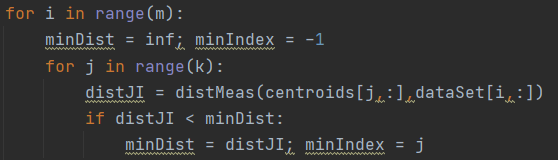
sseNotSplit = sum(clusterAssment[nonzero(clusterAssment[:,0].A!=i)[0],1])

最终找到使得sseSplit+ sseNotSplit最小的簇进行划分。

3.3 k-means(k=2)算法

k-means算法首先随机选取k个点作为初始化簇的中心。用randCent函数实现。

然后对于每个样本点，计算样本点到两个中心的距离，与哪个中心近就划分到哪个簇



全部划分完成之后，更新簇中心。

for cent in range(k):

ptsInClust = dataSet[nonzero(clusterAssment[:,0].A==cent)[0]]

centroids[cent,:] = mean(ptsInClust, axis=0)

3.4 距离计算

在k-means(k=2)和二分k-means算法中，计算距离误差的方法distMeas为球面距离

def distSLC(vecA, vecB):

a = sin(vecA[0,1]\*pi/180) \* sin(vecB[0,1]\*pi/180)

b = cos(vecA[0,1]\*pi/180) \* cos(vecB[0,1]\*pi/180) \*cos(pi \* (vecB[0,0]-vecA[0,0]) /180)

return arccos(a + b)\*6371.0

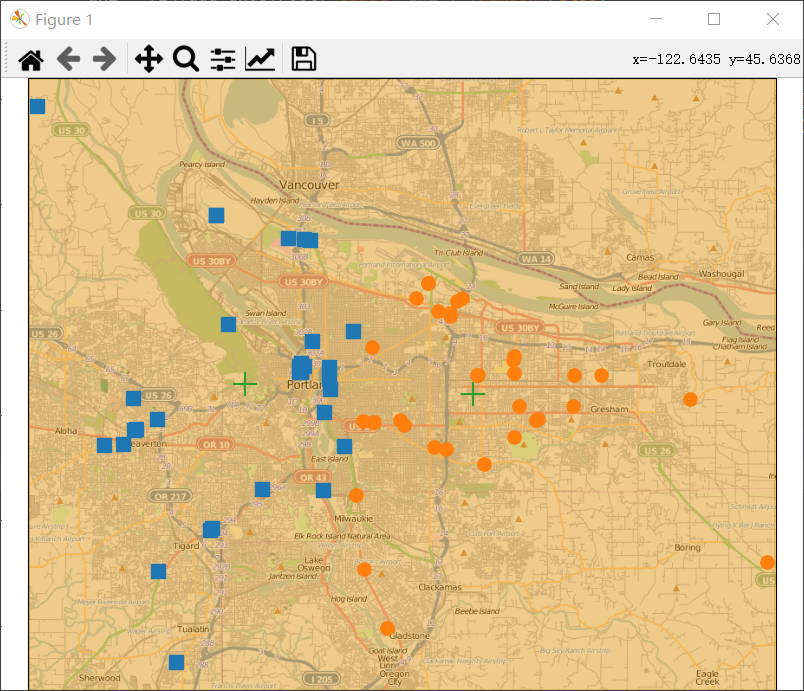
3.5数据可视化

可视化使用matplotlib库，用“+”表示每个簇中心的位置

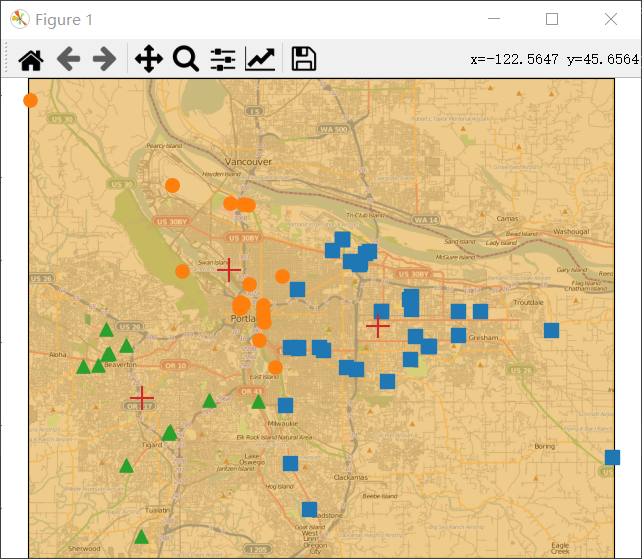
scatterMarkers = ['s', 'o', '^', '8', 'p', 'd', 'v', 'h', '>', '<'] 使不同的数据有不同的显示图形

4.实验结果

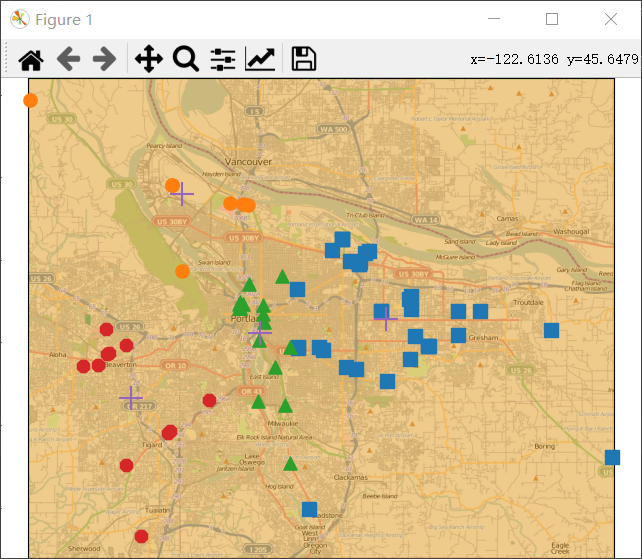
k=2时的结果



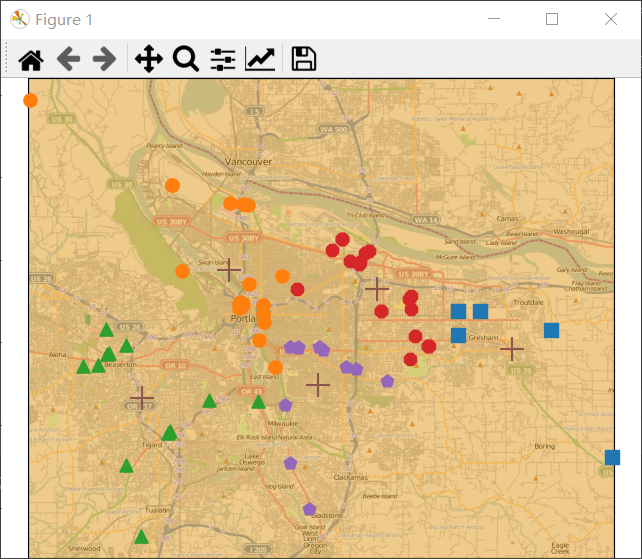
k=3时的结果



k=4的结果



k=5的结果



**任务2：根据用户采集的WiFi信息对用户进行聚类**

1.任务要求：

编写代码分别对DataSetMeans1.csv和DataSetMeans2.csv两个数据集完成聚类实验，k（k>=2）取不同的值，评估聚类的内部指标DB指数，DB指数定义如下：

DB指数（Davies-Bouldin Index，简称DBI）

其中，代表簇C的中心点，avg(C)对应于簇C内样本间的平均距离，对应于簇和中心点间的距离。显然，DBI的值越小越好。

2.实验原理：

2.1 数据集讲解

**数据集：**数据集存于DataSetKMeans1.csv与DataSetKMeans2.csv中，两个数据集相互独立。

**BSSIDLabel:：**SSID标识符，每个AP（接入点，如路由器）拥有1个或多个不同的BSSID，但1个BSSID只属于1个AP；

**RSSLabel：**该BSSID的信号强度，单位dbm；

**RoomLabel:** 该BSSID被采集时所属的房间号；

**SSIDLabel:** 该BSSID的名称，不唯一；

**finLabel：**finLabel标号相同，表示这部分BSSID在同一时刻被采集到；我们将在同一时刻采集的所有BSSID及其相应RSS构成的矢量称为一个指纹

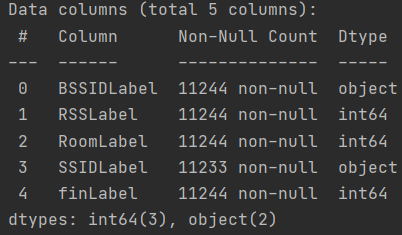
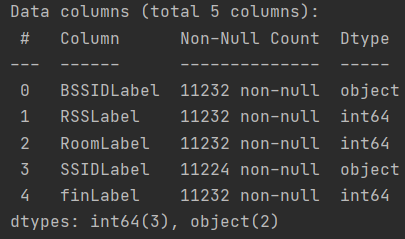
；

由于BSSID的RSS在不同位置大小不同，因此指纹可以唯一的标识一个位置。

本实验采用sklearn.cluster库的KMeans进行训练

3.实验思路

首先观察实验数据。使用pandas库的read\_csv函数读取csv文件，再用DataFrame.info函数得到数据的信息



上图为两个文件的数据信息，可以看出SSIDLabel有少许缺失，其他数据完整。

因为SSIDLabel代表的是WiFi的名称，没有其他用途，所以可直接忽略，不需要做缺失值的处理。

读取数据代码如下，只提取了我们关心的BSSIDLabel, RoomLabel, finLabel, RSSLabel数据

def data\_init(path):

ori = pd.read\_csv(path, encoding='gbk')

bssid\_list = list(set(ori['BSSIDLabel']))

room\_list = ori['RoomLabel'].tolist()

fin\_list = ori['finLabel'].tolist()

rss\_list = ori['RSSLabel'].tolist()

return ori, bssid\_list, room\_list, fin\_list, rss\_list

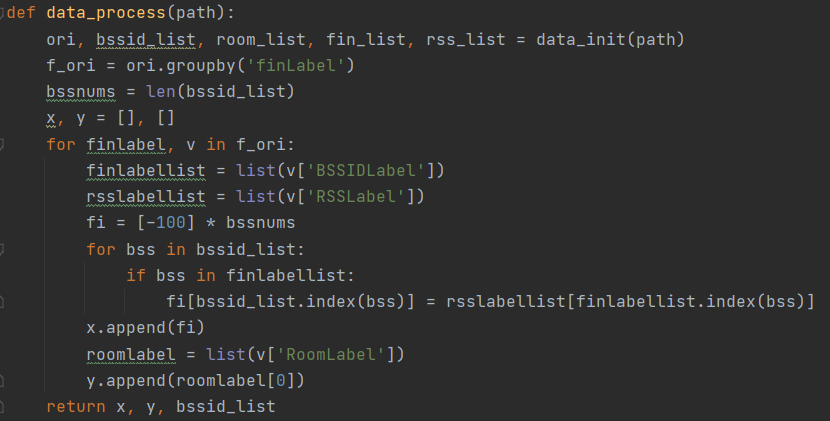
随后对数据进行处理。处理的方式同决策树编程作业时的处理方式。采用特征构造的方法，每一条数据都经过处理变换为一个1行n列的向量,n为数据集中所有出现的mac地址的个数。构建向量有两种方式。

第一种：此条数据中包含的mac地址对应的向量位置的值设置为1，其他值为0。

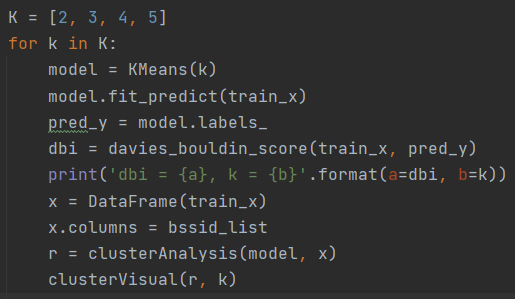
，接收到了1、2、3共三个ID

第二种：此条数据中包含的mac地址对应的向量位置的值设置为对应rss值，其他值为-100

将每一条数据转换而来的向量聚合到一个list中进行训练用，同时每一条数据对应一个房间号，也聚合到一个list中。

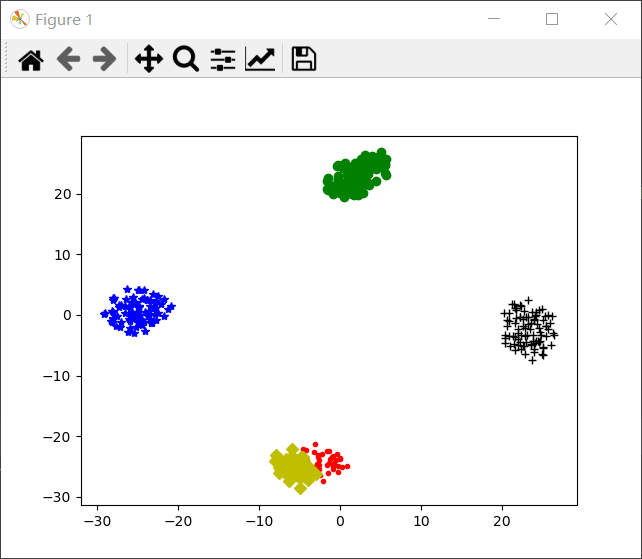
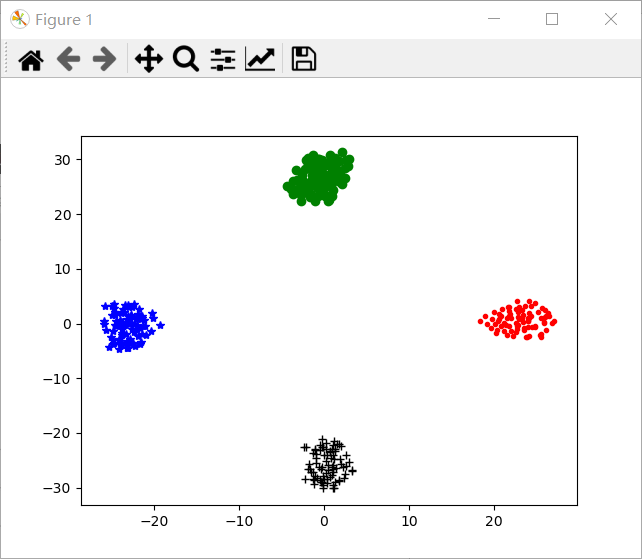
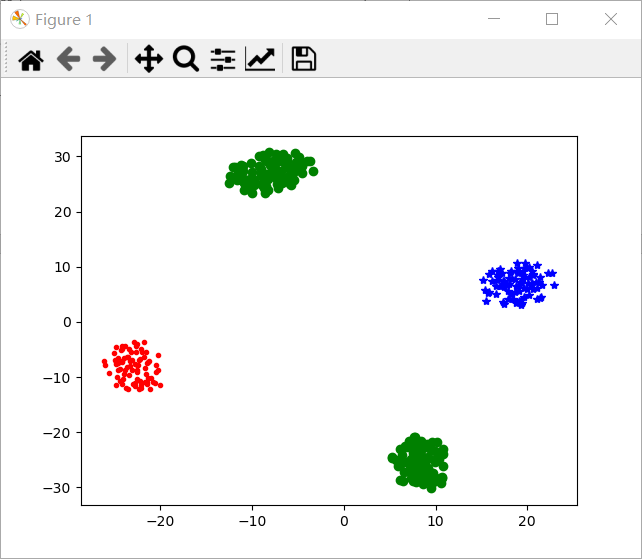
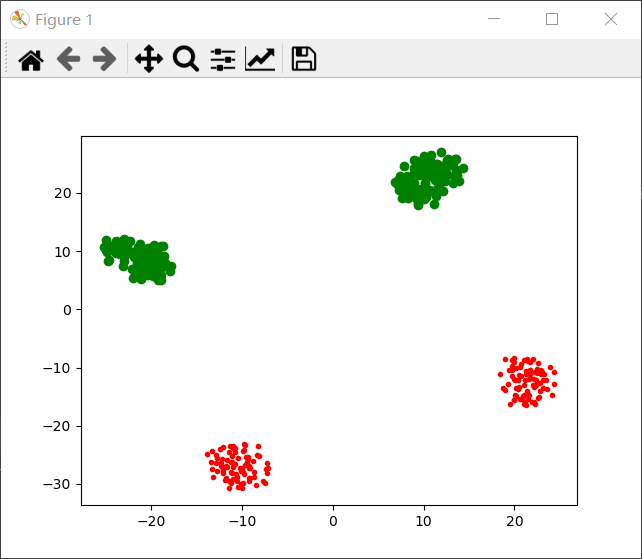


然后用sklearn.cluster库的kmeans算法在不同的k值下进行聚类，并计算dbi，进行可视化。dbi由sklearn.metrics库的davies\_bouldin\_score进行计算。



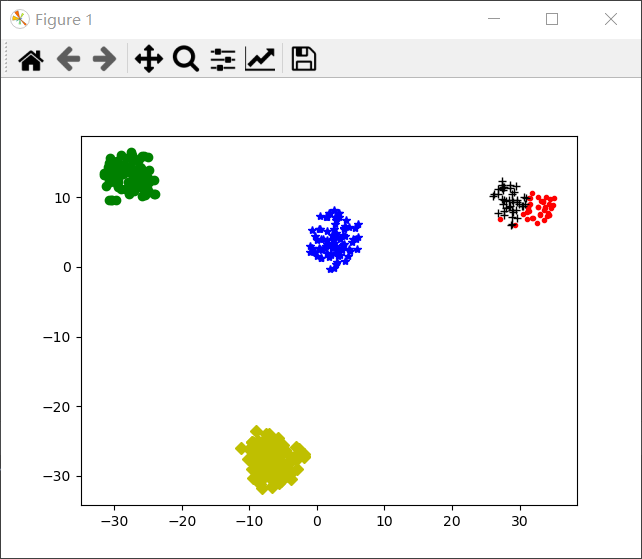
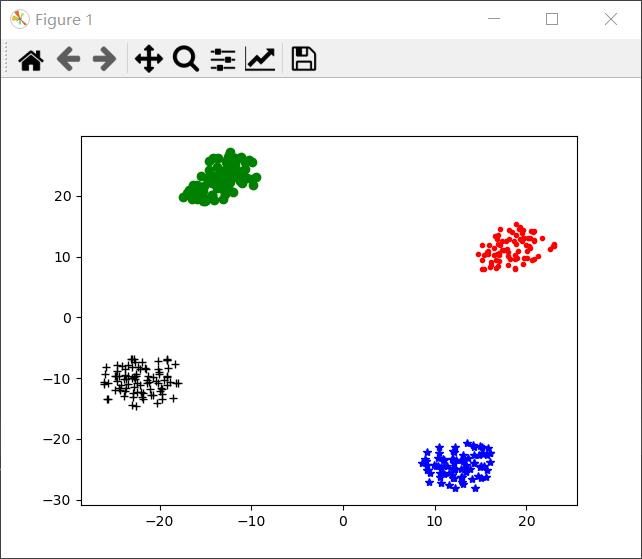
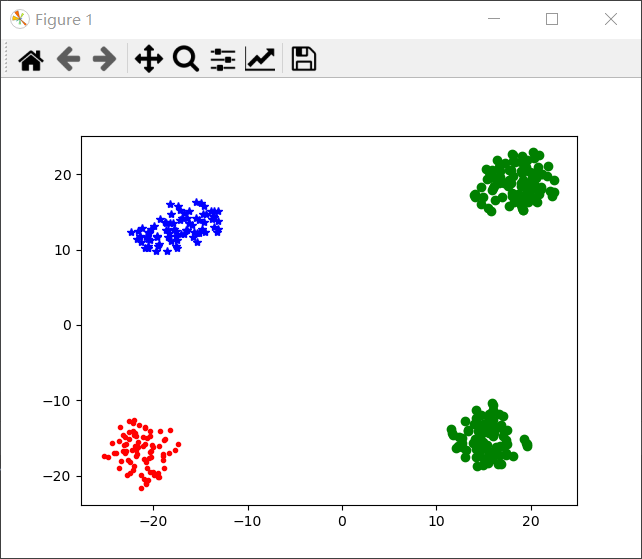
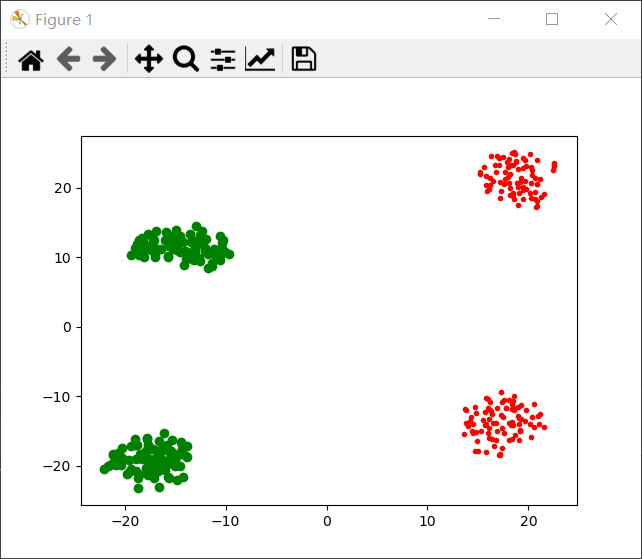
4.实验结果

以下为DataSetKMeans1.csv的不同k值结果



根据dbi判断，最小的dbi为k=2时的聚类，为1.15196

以下为DataSetKMeans2.csv的不同k值结果



根据dbi判断，最小的dbi为k=2时的聚类，为1.146