任务二实验报告

## 任务描述

给定两个人工生成的数据集，需要在这两个数据集上分别实现两个不同种类的聚类方法， 例如用划分聚类（例如K-means等）、密度聚类（例如DBSCAN等）来处理两个数据集。针对两种方法中不同的参数进行实验和结果分析。

## 运行说明

本实验的运行环境为Visual Studio 2012。运行时可以不带命令行参数，表示只对clusters文件夹下已有的聚类簇进行评测，也可以接受1个命令行参数。如果命令行参数为1或3，表示程序会先使用两种聚类算法重新生成数据集1的聚类簇之后再进行评测；如果命令行参数为2或3，表示程序会重新生成数据集2的聚类簇后再评测；如果命令行参数为0，表示程序只对一些特定的参数进行时间复杂度的测试。生成的聚类簇位于程序目录下的文件夹cluster中。

## 算法描述

由于K-means聚类对初始中心的选取较为敏感，因此实验所用的聚类算法为Bisecting K-means（二分K-means）和DBSCAN。

对**Bisecting K-means聚类算法**的描述如下：

1. 选取一个待二分的聚类簇C。

2. 在C上运行基本的K-means聚类算法，将C分解为两个子簇：

2.1. 随机选取C中的两个点作为初始中心点。

2.2. 通过将C中的每个点划分到与之最近的中心点，将C分解为两个子簇。

2.3. 用每个子簇的中心作为新的中心点。

2.4. 重复步骤2.2和2.3，直到中心点不再变化。

3. 重复步骤2若干次（实验中定义ITER\_TIMES为10），选取误差平方和（即SSE）最小的分解方案，用得到的两个子簇取代C。

4. 重复步骤1、2、3，直到聚类簇的总数达到预期。

该算法的步骤1并没有指明如何选取待二分的聚类簇，因此在这一步就有不同的选取方案，例如选取最大的聚类簇、选取平均SSE最小的聚类簇等等。根据参考文献[1]中的说法，这些方案的结果差异很小。实验中对选取最大的聚类簇、选取平均SSE最小的聚类簇两种方案进行了测试，结果如下（purity1和f-score1表示选取最大聚类簇的评价指标，purity2和f-score2表示选取平均SSE最小聚类簇的评价指标）：

由上，可知两种选取方案的差异的确很小，只是在对数据集2中f-score的评测上，选取平均SSE最小的聚类簇稍好一些。因此，实际Bisecting K-means聚类算法中选取的是平均SSE最小的聚类簇。

对**DBSCAN聚类算法**的描述如下：

1. 首先将数据集中的所有点标记为“未访问”。

2. 对于数据集中每一个标记为“未访问”的点P：

2.1. 将点P标记为“已访问”。

2.2. 计算点P的邻域N，如果N中点的数目小于minPts，将点P标记为“噪声点”，转到步骤3；否则，创建一个新的聚类簇C，将点P添加到C中。

2.3. 对于N中的每个点：

2.3.1. 如果点标记为“未访问”，将点标记为“已访问”，并计算点的邻域，若中点的数目不小于minPts，则将中的点添加到N中；

2.3.2. 如果点不是任何聚类簇的成员，将其添加到C中。

2.4. 重复步骤2.3，直到N为空。

3. 重复步骤2，直到数据集中不存在“未访问”的点。

## 参数分析

对**数据集1**而言，考虑其人工标注簇的数目、数据规模、数据点在每一维上的分布等因素，将其参数确定如下：

Bisecting K-means中的k：const int para1\_k[KNUM] = {11, 13, 15, 17, 19};

DBSCAN中的eps：const double para1\_eps[PNUM] = {3e4, 3.5e4, 4e4, 4.5e4, 5e4};

DBSCAN中的minPts：const int para1\_minPts[PNUM] = {50, 75, 100, 125, 150};

**Bisecting K-means**的评测结果如下：

由上图，可知当k增大时，指标purity和f-score的取值倾向于更好，但这并不一定表示其聚类效果更好（对数据集1而言，若取k = 数据量 = 5000，则必有purity = 1，然而这样的聚类并不科学）。

**DBSCAN**的评测结果如下：

纵向比对以上各图，可得当minPts固定、eps变化时purity和f-score的变化情况。由上，可知当eps固定、minPts递增或者minPts固定、eps递减时，purity基本呈上升态势（末端有波动但不明显），而f-score明显有先增后降的趋势。当然这也与参数的选取有关，例如在eps较小或较大、minPts递增时，f-score基本是单调的。综合来看，当eps = 4e4、minPts = 100时，purity和f-score的取值是最好的，此时聚类簇的数目为15，恰好等于人工标注簇的数目。

对**数据集2**而言，考虑其人工标注簇的数目、数据规模、数据点在每一维上的分布等因素，将其参数确定如下：

Bisecting K-means中的k：const int para2\_k[KNUM] = {5, 6, 7, 8, 9};

DBSCAN中的eps：const double para2\_eps[PNUM] = {0.7, 0.8, 0.9, 1.0, 1.1};

DBSCAN中的minPts：const int para2\_minPts[PNUM] = {3, 4, 5, 6, 7};

**Bisecting K-means**的评测结果如下：

与数据集1类似，当k增大时，指标purity和f-score的取值倾向于更好。

**DBSCAN**的评测结果如下：

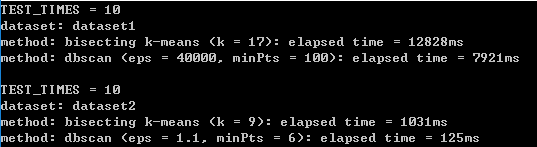
与数据集1类似，当eps固定、minPts递增或者minPts固定、eps递减时，purity基本呈上升态势，而f-score仍有先增后降的趋势，但已不太明显。综合来看，当eps = 1.1、minPts = 6时，purity和f-score的取值是最好的，此时聚类簇的数目为11，与人工标注簇的数目相近。

## 时间和聚类效果分析

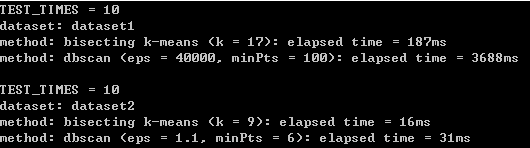
由以上各图，不难发现在一定的参数选取下，使用purity和f-score作为评价指标，DBSCAN的最佳聚类效果优于Bisecting K-means的最佳聚类效果。假定purity和f-score具有相同的权重，记S = purity + f-score。在数据集1中，使用Bisecting K-means聚类，当k = 17时，；使用DBSCAN聚类，当eps = 4e4、minPts = 100时，。在数据集2中，使用Bisecting K-means聚类，当k = 9时，；使用DBSCAN聚类，当eps = 1.1、minPts = 6时，。

输入命令行参数0，对上述各参数进行时间复杂度的测试，结果如下（每种聚类重复运行10次，测试时生成的聚类结果不输出到文件）：

Debug模式：



Release模式：



由上，可知在Release模式优化之后，聚类效果较好且相近时，Bisecting K-means的时间复杂度优于DBSCAN。

## 参考文献

[1] Steinbach, Michael, George Karypis, and Vipin Kumar. "A comparison of document clustering techniques." KDD workshop on text mining. Vol. 400. No. 1. 2000.