# 数据挖掘中基于密度的聚类分析算法

王劲波[1],翁 伟[2],许华荣[3]
(1.厦门大学计划统计系,福建 厦门 361005;2.厦门理工学院电工系,福建 厦门 361005;
3.福建师范大学福清分校,福建 福清 50300)

摘 要:聚类在数据挖掘、模式识别等许多领域有着重要的应用。本文介绍了聚类算法的几种分类,并例举了几种基于密度的聚类算法。最后以一种新颖的基于最大不相含核心点集的聚类算法LSNCCP为例,详细介绍整个聚类算法的工作过程。

关键字:数据挖掘;聚类;密度;核心点;最大不相含核心点集
中图分类号:C812 文献标识码:A 文章编号:1002-6487(2005)10-0139-03

## 0 前言

数据挖掘(Data Mining)就是从大量的、不完全的、有噪声的、模糊的、随机的数据中发现隐含的、先前未知的、对决策有潜在价值的规则的过程[1]。从上世纪七十年代开始到至今,在数据挖掘领域的研究已经取得了相当丰硕的成果,并产生了很多应用于实际商业活动的系统,这些系统为企业界、政府组织等带来了巨大的效益。

聚类(Clustering)是数据挖掘领域中最活跃的研究分支之一,聚类在统计学、模式识别、图像处理、机器学习、生物学、市场营销等许多领域有广泛的应用。所谓聚类,就是将物理或抽象对象的集合组成为由类似的对象组成的多个类或簇(cluster)的过程,由聚类所生成的簇是一组数据对象的集合,同一簇中的对象尽可能相似,而不同簇中的对象尽可能相异。通过聚类,人们能够识别密集的和稀疏的区域,发现全局的分布模式和数据属性之间有趣的相互关系,如在商业上,聚类可以帮助市场分析人员从消费者数据库中区分出不同的消费群体来,并且概括出每一类消费者的消费模式或者说习惯。在数据挖掘中,聚类分析能作为一个独立的工具来获得数据分布的情况,观察每个簇的特点,集中对特定的某些簇做进一步的分析。此外,聚类分析还可以作为其它算法(如特征和分类等)的预处理步骤,这些算法再在生成的簇上进行处理。聚类分析已经成为数据挖掘领域一个非常活跃的研究课题。

## 1 几种具有代表性的基于密度的聚类算法

### 1.1 DBSCAN算法

DBSCAN[2](Density-Based Spatial Clustering of Applications with Noise)是一个基于高密度连接区域密度聚类算法。这个方法将密度足够大的那部分记录组成聚类,其基本思想涉及一些新的定义。

定义1 对于给定的对象,我们称在其半径ε范围内的一个记录为这个记录的ε-邻域。

定义2 如果一个对象的ε-邻域个数超过一个最小值MinPts,那么我们就将这个记录称作核心对象。

定义3 一个对象的集合D,我们说一个对象p在q的ε-邻域内,且q是一个核心对象,我们说对象p是从对象q出发直接密度可达的。

定义4 一个对象链$p_1,p_2,\cdots,p_n$,如果$p_1=q,p_n=p$,对$p_i\in D,(1\leqslant i\leqslant n)$。$p_{i+1}$是从$p_i$出发的关于ε和MinPts直接密度可达的,则对象p是从对象q关于ε和MinPts密度可达的。

定义5 如果对象集合中存在一个对象o,使得对象p和对象q是从o关于ε和MinPts密度可达的,那么对象p和对象q是关于ε和MinPts密度相连的。

DBSCAN通过检查所给数据集合当中每个点的Eps邻域来寻找聚类。首先它从第一个点开始,如果它是关于Eps和MinPts的核心点就创建一个新的类,然后DBSCAN反复从这些核心点寻找可以直接密度可达的点,这个过程可能会涉及一些密度可达类的合并,至到没有新的点可以归入此类。如果第一个点不是核心点,就先把它暂时标志成为噪声点,然后跳到下一个点。对于下一个点,如果它是核心点而且没有被分入某一类,那就像对待第一个点一样创建一个新的类,并把那些从它可以密度可达的点全部加入这一类中。这个过程只到没有新的点可以添加到任何类的时候结束。该算法的时间复杂度为$O(n^2)$。

### 1.2 OPTICS算法

OPTICS[3](Ordering Points to Identify the Clustering Structure)是通过对象排序识别聚类结构的聚类算法。当采用DBSCAN进行聚类时,参数ε和MinPts是由用户输入,而对于真实的高维数据集合而言,参数设置的细微不同可能导致差别很大的聚类结果。为了解决这个难题,提出了OPTICS。在该算法中,为自动和交互的聚类分析计算一个簇次序,而

这个次序代表了数据的基于密度的聚类结构。在该次序中选择根据最小的 $\varepsilon$ 值密度可达的对象，以便高密度的聚类能被首先完成。

基于这个思想，在 OPTICS 中，对于核心对象需要另外存储两个值：核心距离和可达距离。核心距离是指使得 p 成为核心对象的最小 $\varepsilon$。可达距离是指核心对象 p 的核心距离和 p 与对象 q 的欧几里得距离之间的较大值。通过额外存储的这两个值，OPTICS 算法创建了数据库中对象的一个次序。在为提取所有基于密度的聚类过程中，对于小于在生成该次序中采用的距离 $\varepsilon$ 的任何距离 $\varepsilon'$，这些信息是足够的。

由于 OPTICS 算法具有和 DBSCAN 算法在结构上的等价性，两者具有相同的时间复杂度 $O(n^2)$。

1.3 FDBSCAN 算法

当数据量相当大时，DBSCAN 的时间复杂度是相当高的。如果我们能够降低区域查询的次数，我们就可以改进 DBSCAN 的时间复杂度。对于密集的类来说，在一个核心点的邻域中有相当多的点可以不用作为类扩展用的核心点，因为一个点的 Eps 邻域的点通常是被其他的点的 Eps 邻域所覆盖。为了减少 DBSCAN 的时间复杂度，应该选择一个点的 Eps 邻域内的部分点来做进一步扩展的核心点，FDBSCAN[4] (Fast Density-Based Spatial Clustering of Applications with Noise) 算法正是基于这样的想法提出的。对于二维空间数据，FDBSCAN 算法建议核心点周围 4 个代表对象进行类扩展；对于三维空间数据，建议选择 6 个代表对象；依次类推，在 n 维空间中，选择 2n 个代表对象。也就是说，在每一维空间上，选择两个对象作为代表对象用于类的扩展。FDBSCAGN 算法选出一个与核心点最远的点作为第 1 个代表点；随后则选出离所有已被选出的代表点最远的点作为下一个代表点，直到选出所需的全部代表点为止。

FDBSCAN 进行聚类时的丢失点比较少，可以不做处理，FDBSCAN 的时间复杂度也为 O(n*logn)，与 DBSCAN 比较起来，它不需要空间索引树的支持。

1.4 IGDCA 算法

文献[5]给出另外一个基于密度的增量算法 IGDCA (Incremental Grid Density-Based Clustering Algorithm)，它把需要聚类的数据空间分成大小相等的网格，然后对密度超过阈值的网格进行用 DBSCAN 进行聚类，这个算法存在两个缺陷：在划分网格的时候是基于对要聚类数据空间进行划分，但是如果这个范围所选过大，则要聚类的网格数量非常可观，如果所选范围太小，很有可能出现更新的数据超出所选范围；在进行增量更新时，如果是删除操作，那 IGDCA 只是调用它的非增量算法进行处理，如果是插入操作，在有大数据量进行更新时，跟非增量算法比较起来，它所获得的收益也很有限。

## 2 聚类算法实例

2.1 LSNCCP 算法简介

文献[6]中提出了一种 LSNCCP 算法 (the Largest Set of Not-Covered Core Points)。该算法是一种基于最大不相含核心点集的聚类算法，属于基于密度的聚类算法。LSNCCP 算法中，在定义密度的基础上，通过考察核心点之间的三种关系，进而选出所有核心点的一个子集，该子集满足特定的关系，称为最大不相含核心点集，在此基础上进行聚类。该算法能高效的产生聚类并且不丢失任何可以聚类的点。实验结果证明，该方法在速度和精度方面都有非常高的效率，特别适用于高密度大数据量的聚类。本文的内容结构安排如下：首先介绍相关的概念，然后详细给出算法描述，然后给出实验结果与分析，说明该算法的可行性和优越性。

在 LSNCCP 算法中定义了核心点之间的三类不同的关系：

定义 1 如果两个核心点 p，q，有 Dist(p，q)<=Eps，则称 p，q 互为相含核心点(cover core points)。

定义 2 如果两个核心点 p，q，有 Eps<Dist (p，q)<=2*Eps，则称 p，q 互为相交核心点( intersectant core points)。

定义 3 如果两个核心点 p，q，有 Dist(p，q)>2*Eps，则称 p，q 互为相离核心点(separate core points)。

为了下文算法描述的方便性，用 Eps 表示设定的半径。另外，设定一个密度阈值 MinPts。同时我们用 Add(S,p)表示把点 p 加到集合 S 中去，用 Delect (S，p)表示把点 p 从集合 S 中删除。

算法首先要求出一个最大不相含核心点集，把每个核心点都看成是一个小类的中心建立一个个单独的类，然后如果有两个核心点是相交核心点，则合并这两个小类为一个大类；同理，如果一个核心点与一个大类中某个核心点是相交核心点，则也相合并。这就是对最大不相含核心点集的分类和重新建立数据到类的映射的过程。然后找回丢失点，正确聚类到某一个类中，最终完成聚类。

2.2 最大不相含核心点集选取算法

本步骤实现这样的目标：找出所有核心点的一个子集 ReferenceSet，在该子集中任何两个点均不相含，并且不存在其它的核心点 r，$r \notin$ ReferenceSet，且 r 与集合 ReferenceSet 中任何点都不相含。不妨称这种集合为最大不相含核心点集。

这一步骤又细分为两个步骤：首先找出那些可能成为核心点的点，这些点不妨称这些点为候选核心点；然后进一步判断这些点是否是核心点，并同时建立数据到核心点的映射。具体算法如下所示：

```
GenerateReference ( PointSet,Eps,MinPts,ReferenceSet )
For i=1 to PointSet.size {
If dist (ReferenceSet,Pi)>Eps then
Add (ReferenceSet,Pi);
End If}              // 从有待聚类数据中选取候选核心点
For j=1 to ReferenceSet.size {
Result=PointSet.RegionQuery (Pj,Eps);
If  Result.Size>=MinPts then
                    // 取核心点
PointSet.ChangeClIds (Result,j);
```

// 对核心点的代表区域看成是一个小类，做数据到该核心点的映射

Else Delete (ReferenceSet，$P_j$)；

// 在集合 ReferenceSet 中删除该非核心点

}　　// 对集合 ReferenceSet 中的所有点进行区域查询

For k=1 to PointSet.size {

If Pk.ClIds=0 and dist (ReferenceSet，Pk)>Eps then {

// 取没有与核心点集中任何一个点作映射的当前点

Result= PointSet.RegionQuery (Pk，Eps)；

if Result.Size>=MinPts then {

// 判断该点是否为核心点

PointSet.ChangeClIds (Result，k)；

Add (ReferenceSet，Pk)；}

// 把该点加进核心点集合之中，并做数据到该点的映

}}　　// 生成一个最大不相含的核心点集

**算法 1　最大不相含核心点生成并建立数据到核心点的映射**

### 2.3　数据聚类和丢失点处理

当算法 1 运行结束以后，将生成一个最大不相含的核心点集，接下来就要将数据根据所得到的最大不相含的核心点集进行聚类。开始我们把最大不相含核心点集中的每一个点 $P_i$ 看成是一个簇 $C_i$，即 Ci= {$P_i$} 然后，如果簇 $C_i$ 中有一个点 p 与簇 $C_j$ 中一个点 q 互为相交核心点，则合并这两个簇成一个簇 C，既 $C=C_i \cup C_j$。然后，对输入的所有数据，逐一查看每一个点与某个簇中的核心点的距离，一旦发现距离少于等于 Eps，即分配到该簇中。这个过程结束后得到的簇的个数就是整个数据可以分出的聚类数。

当采用最大不相含核心点集中的核心点来聚类的话，会有一些本该聚类的点得不到聚类，也就是说聚类结果会丢失一些点。丢失点一定不是核心点。故可以把一个没有得到聚集的点的代表区域内的所有的点集合　距离少于等于 Eps）与已经得到聚类的点的集合作一个交集，在判断这个交集中的点是否是核心点。一旦判断出一个点是核心点，就说明这个点是丢失点，我们就把这个丢失点聚集到这个核心点同一个类中去。如果这个交集为空，或者交集中没有核心点，则这个点是噪音。对于噪音点，将被标记出来，最终形成噪音点集合，该集合可用作噪音分析以及孤立点分析。

### 2.4　实验分析

首先，利用本文提出的算法，对图 1 中 (a)的 1100 个点进行聚类，聚类结果在图 2 的 (b)和 (c)中。(b)表示得到的一个最大不想含核心点集，共有 99 个点；(c)为聚类结果，共有 933 个点。从结果中可以看出，该算法的结果跟人的直觉一样，能够进行正确的聚类。从图 1 中也可以看出，本算法对聚类的形状不敏感，能够发现任意形状的聚类，能有效的排除噪音。

(a) 数据集　　(b)　个最大不相含核心点集　　(c) 聚类结果

图 1　1000 个点及其聚类结果

该算法性能与数据记录规模(n)没关系，而与密度有关系，当密度越大时，其性能越好，优越性越高。


参考文献：

[1]Fayyad M., Piatetsky-Shapiro G., Smyth P., etc., From data mining to knowledge discovery: an overview[C], Advances in Knowledge Discovery and Data Mining Menlo Park, CA, 1996.

[2]Ester M., Kriegel HP, Sander J., Xu X., A density based algorithm for discovering clusters in large spatial databases with noise [C]. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, 1996.

[3]Ankerst M., Elsen C., Ester M., and Kriegel H.-P., Visual classification: An interactive approach to decision tree construction [C]. In Proceedings of 1999 International Conference on Knowledge Discovery and Data Mining, San Diego, 1999.

[4]周水庚，周傲英，曹晶，胡运发，一种基于密度的快速聚类算法[J]. 计算机研究与发展，2003，37(11).

[5]CHEN Ning, CHEN An, ZHOU Long-xiang, An Incremental Grid Density-Based Clustering Algorithm[J]. Journal of software, 2002, 13(1).

[6]翁伟，薛永生，文娟，王劲波.一种基于最大不相含核心点集的聚类算法[J].计算机研究与发展，2004，(1).


责任编辑/易永生）