文章编号: 1001-0920 (2013) 06-0815-08

基于相对密度的混合属性数据增量聚类算法

黄德才, 李晓畅

(浙江工业大学 计算机科学与技术学院, 杭州 310023)

摘 要: 传统的基于密度的带噪声空间数据聚类算法主要存在以下问题: 聚类只对具有数值属性的数据有效, 而对具有非数值属性的数据失效; 参数设置困难且聚类结果对参数较为敏感; 聚类的度量以绝对密度值为标准, 无法发现密度等级不同的聚类结果. 针对以上问题, 提出一种面向混合属性数据的、基于相对密度的聚类算法 RDBC_M, 同时提出解决这类问题的增量式聚类算法, 并从理论和仿真实验两方面分析、验证了算法的有效性和加速效果.

关键词: 相对密度; 绝对密度; 混合属性数据集; 增量聚类

中图分类号: TP311 **文献标志码:** A

Incremental relative density-based clustering algorithm for mixture data sets

HUANG De-cai, *LI Xiao-chang*

(College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China. Correspondent: HUANG De-cai, E-mail: hdc@zjut.edu.cn)

Abstract: Traditional density-based clustering algorithm mainly has three problems as follow. Firstly, it only supports spatial attributes without considering non-spatial attributes in the database. Secondly, it is difficult to set the parameters, and the clustering result is sensitive to the parameters. Thirdly, it can't discover the clusters of different density for adopting absolute density as the metrics of all clusters. In order to overcome these problems mentioned above, the paper presents an relative density-based clustering algorithm for mixture data sets(RDBC_M), and further carries out the research on its incremental clustering algorithm. Theoretical analysis and simulation experiment verfy the effectiveness and the performance speed-up effect of the proposed algorithm.

Key words: relative density; absolute density; mixture data sets; incremental clustering

0 引 言

聚类分析是数据挖掘领域的一项重要研究内容, 可以作为一个独立的工具来获得数据分布的情况[1]. 所谓聚类是指将一个给定的数据集划分成若干个称为类或簇的子集, 划分的原则是使在同一个簇中的对象之间具有较高的相似度, 而不同簇中的对象之间差别较大. 迄今为止, 人们已提出了许多聚类算法, 其中, 基于密度的带噪声空间数据聚类算法 DBSCAN[2]是一个较为典型的基于密度的聚类算法, 解决了传统基于距离的算法仅能找到"类圆形"簇的缺陷. 但 DBSCAN 存在 3 个主要缺陷[3]: 1) 仅适用于数值属性而不适于分类属性; 2) 聚类参数 ϵ 和 MinPts 的设置较为困难; 3) 聚类参数值 ϵ 和 MinPts 使用全局恒定值, 无法得到不同密度等级的聚类结果.

聚类的本质是使得一个簇中的所有对象具有较高的相似度, 即同一个簇中对象之间的密度相似. 容易理解的是, 簇与簇之间的界线是相互比较而划分的, 密度差异是相对的. 因此文献 [4] 引入了相对密度的概念, 提出了基于相对密度的 RDBClustering 聚类算法. 该算法能发现任意形状、不同密度等级的簇, 但对具有非数值属性的数据集失效, 且相对密度的计算公式误差较大. 文献 [5] 面向移动数据和轨迹数据提出基于相对密度的 k 近邻聚类算法 RDCTD, 该算法利用 DBSCAN 中核心对象的概念, 首先依据参数 ε 和 MinC 找出核心点和初始簇, 然后进行聚类扩展, 与传统 DBSCAN 算法不同的是, 在聚类扩展时引入了 k 近

收稿日期: 2012-02-28; **修回日期:** 2012-06-06.

基金项目: 国家重大科技专项专题项目(2009ZX07318-003-01).

作者简介: 黄德才(1958–), 男, 教授, 博士生导师, 从事数据挖掘、人工智能等研究; 李晓畅(1983–), 男, 硕士生, 从事数据挖掘的研究.

邻居的概念. 该算法为移动数据的聚类和交通状态分析提供了一种新途径, 但依然要用到DBSCAN中的参数, 因此, RDCTD聚类算法也存在DBSCAN中对参数敏感的问题. 有些基于距离的聚类算法也引入相对密度的概念, 如文献[6]提出了两个基于相对密度权值的FCM(relative density weights based fuzzy C-means)聚类算法, 算法对FCM进行了改进, 但依然没有克服FCM和K-means中存在的问题, 如聚类结果受初始值的影响、不同初始对象的选择对目标函数收敛影响较大、对于海量数据时很难得出可靠的聚类结果.

在增量聚类算法的研究方面, 增量聚类算法面对的数据集是动态变化的, 数据集的密度也在不断变化, 无法采用全局唯一的绝对密度参数对数据集进行划分, 因此一些传统的基于密度的增量聚类算法难以有效适用, 如Incremental DBSCAN等[7-9]增量聚类算法均存在这类问题. 针对基于相对密度的增量聚类算法, 目前开展的研究较少, 文献[9]提出了一种基于相对密度的增量聚类算法RDBIClustering, 但研究尚不够全面和深入, 主要存在以下不足: 1) 只考虑了插入情况, 未考虑删除情况下的聚类变化; 2) 只说明了插入操作有孤立点、创建新类、吸收、合并和拆分等5种不同处理情况, 未详细分析何种情况下会出现这些聚类处理; 3) 未对原聚类算法和增量聚类算法进行性能比较, 以表明增量聚类的性能优势. 因此, 基于相对密度的增量聚类算法依然有待更深入和细致的研究.

鉴于此, 本文在对RDBClustering聚类算法进行改进的基础上, 提出基于相对密度的混合属性数据聚类算法RDBC_M, 并在RDBC_M算法的基础上, 进一步研究了其增量聚类算法, 提出基于相对密度的混合属性数据集增量聚类算法IncRDBC_M. 增量算法的研究工作是在分析如下问题中逐步展开的: 当有某个对象p插入或删除时, 哪些对象的纯核心对象地位会发生变化, 并由此确定其影响集; 然后在此基础上提出增量聚类的方法, 逐一分析插入或删除对象时的增量聚类方法; 最后, 利用二维和三维空间模拟数据进行实验, 分析和验证了算法的有效性和加速性能. 该算法与其他增量聚类算法的最大不同在于可以发现密度等级不同的簇.

1 基于相对密度的混合属性数据聚类算法

1.1 相关定义

传统的距离度量公式仅能表示数值属性的差异信息, 面向维度的距离[11]是可以同时表示数值差异和维度差别的距离度量方法. 文献[13]推广了这一概念, 并针对混合属性对象提出了相似度的概念. 本文利用传统欧氏距离, 重新定义了混合属性数据的距离公式和相似度函数.

设D为n维数据(对象)集, $p,q,o\in D$; 对象p和q的第i维数据分别用x_i和y_i表示, 且x_i和y_i均可以是数值属性或分类属性值.

定义1 设$p,q\in D$为两个具有数值属性和分类属性的对象, 称

$$M\_\mathrm{Dist}(p,q)=\sqrt{\sum_{i=1}^{n}\mathrm{dist}^2(x_i,y_i)} \tag{1}$$

为混合属性对象p和q之间的欧氏距离, 简称混合属性距离.

由定义1可知, 只要对不同类型的属性使用不同的距离公式dist, 就可以用$M_$Dist同时表示分类属性和数值属性的距离度量. 如, 当第i维是数值型属性时, 可以使用$\mathrm{dist}\,(x_i,y_i)=|x_i-y_i|$作为距离函数; 当第$i$维是具有$m$个值$C_1$,$C_2$,$\cdots$,$C_m$的分类属性时, 可以引进一个$m\times m$的矩阵, 以存储该维度不同取值之间的相似度, 即

$$\begin{bmatrix} 0 & & & & \\ d_{21} & 0 & & & \\ d_{31} & d_{32} & 0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ d_{m1} & d_{m2} & d_{m3} & \cdots & 0 \end{bmatrix}.$$

其中$d_{ij}(i,j=1,2,\cdots,m)$的值是分类属性值C_i和C_j的相似性度量, 通常由熟悉该领域的专家经过综合评价后赋值. 相似度d_{ij}越接近0表示C_i和C_j这两个簇越相似. 由此, 分类属性$x_i=C_i$和$y_i=C_j$的相似度为

$$\mathrm{dist}(x_i,y_i)=\mathrm{dist}(C_i,C_j)=d_{ij}. \tag{2}$$

为了使新算法适用于分类属性, 将文献[12]中的最近邻居距离概念予以推广, 提出纯近邻的定义.

定义2 对于给定的正整数k, 对象p的k纯近邻距离记为$k_$purdist(p), 定义为p到其第k个最近邻居o的混合属性对象的欧氏距离$M_$Dist(p,o).

定义3 对象p的k纯近邻邻居记为$N_{\mathrm{pure}}(p,k)$, 定义为集合

$$N_{\mathrm{pure}}(p,k)=$$

$$\{q\in D-\{p\}|M\_\mathrm{Dist}(p,q)\leqslant k\_\mathrm{purdist}(p)\}.$$

对象p的k个最近纯邻居即为集合$N_{\mathrm{pure}}(p,k)$, 其中所有对象与对象p互为纯邻居, 因此, 它们在分类属性上是相似的.

本文以$M_$Dist作为距离的度量函数, 并对文献

[9] 中的相对密度概念进行推广, 由此给出定义 4.

定义 4　对象 p 关于其 k 个纯近邻的相对密度为

$$rd_{N_{\mathrm{pure}}(p,k)}(p)=\frac{1}{k}\sum_{o\in N_{\mathrm{pure}}(p,k)}\left|\frac{\mathrm{npnd}(o,k)}{\mathrm{npnd}(p,k)}-1\right|, \quad (3)$$

其中 $\mathrm{npnd}(p,k)$ 是对象 p 与 k 纯近邻的绝对密度, 一般可以定义为 p 与其 k 个纯邻居的距离平均值.

式 (3) 改进和优化了文献 [9] 中相对密度的计算公式, 其公式表示同一个簇中所有对象与核心对象之间密度比的平均值, 而式 (3) 更准确地表示了同一个簇中对象之间密度的平均差值.

类似文献 [9] 中核心对象的概念 (相对密度小于给定阈值 λ 的对象), 对于聚类对象包含分类属性的数据集引入相应的概念.

定义 5　给定阈值 $0<\lambda<1$, 若 $rd_{N_{\mathrm{pure}}(p,k)}(p)<\lambda$, 则称对象 p 为纯核心对象.

为了使新算法可以用于不同密度等级的聚类, 对文献 [2] 中密度可达、直接密度可达和密度相连的有关概念也进行推广和改进, 分别引进直接相对密度可达和相对密度接连的概念.

定义 6　给定阈值 $0<\lambda<1$ 和一个对象链 $p_1, p_2, \cdots, p_n$, 其中 $p=p_1, q=p_n$. 对于任意 $p_i\in D(i=1,2,\cdots,n)$, 存在:

1) 若 p_i 是纯核心对象, 且其 k 个最近的纯邻居包含 p_{i+1}, 则称对象 p_{i+1} 从 p_i 出发是关于阈值 λ 直接相对密度可达的, 记作 $p_i\rightarrow^{\lambda}p_{i+1}$;

2) 若存在 $p\rightarrow^{\lambda}q$ 或 $q\rightarrow^{\lambda}p$, 则称对象 p 与 q 关于阈值 λ 相对密度接连, 记作 $q\leftrightarrow^{\lambda}p$.

定义 7　若对象集 CorSet 中的对象满足下列 3 个条件, 则称其为一个纯核心对象集:

1) 对于给定的阈值 $0<\lambda<1$, $\forall p\in$ CorSet, 均有 $rd_{N_{\mathrm{pure}}(p,k)}(p)<\lambda$, 即 CorSet 中的每个对象均为纯核心对象;

2) $\forall p,q\in$ CorSet 均有 $q\leftrightarrow^{\lambda}q$, 即 CorSet 中的对象是相对密度接连的;

3) $\forall p\in$ CorSet, 记 $\overline{\mathrm{CorSet}}=\mathrm{CorSet}-\{p\}$, 均有 $rd_{\overline{\mathrm{CorSet}}}(p)<\lambda$, 其中

$$rd_{\overline{\mathrm{CorSet}}}(p)=\frac{1}{|\overline{\mathrm{CorSet}}|}\sum_{o\in\overline{\mathrm{CorSet}}}\left|\frac{\mathrm{npnd}(o,k)}{\mathrm{npnd}(p,k)}-1\right|. \quad (4)$$

定义 7 中, $rd_{\overline{\mathrm{CorSet}}}(p)$ 是集合 CorSet 中对象 p 与集合中其他对象密度差距的平均值, 其值的大小刻画了 p 与集合中其他对象的密度差距大小, 值越小, 表明 p 越能与集合中的其他对象融为一体. 式 (4) 与文献 [4,10] 中的相对密度公式相比, 密度误差更小, 后者代表的是同一个簇中所有对象与核心对象之间密度比的平均值, 式 (4) 刻画的是同一个簇中对象之间密度的平均差异值, 其相对密度值更加准确.

定义 8　设 CorSet 为一个纯核心对象集, 若 C 是下述两部分的集合: CorSet 和从 CorSet 中任一对象出发构成直接相对密度可达关系的所有对象, 则称 C 为 CorSet 对应的一个聚类或一个簇.

根据定义 8, 对于给定的阈值 $0<\lambda<1$ 和在对象集 D 中的某一纯核心对象集 CorSet, 存在 CorSet 对应的簇 $C=\mathrm{CorSet}\bigcup\{p|\exists o\in\mathrm{CorSet}, \text{有 } o\rightarrow^{\lambda}p\}$.

1.2　基于相对密度的混合属性数据集聚类算法

本文以相对密度而非绝对密度作为聚类计算准则, 将相对密度低于阈值 λ 的所有对象作为核心对象, 并将相对密度接连的核心对象及其近邻对象构成的集合作为一个聚类或一个簇, 由此提出基于相对密度的面向混合属性数据集的聚类算法 RDBC_M, 计算步骤如下.

Step 1: 从数据集 D 中取出一个未经处理的点 p.

Step 2: 如果点 p 是纯核心对象, 则以 p 为起始点生成一个纯核心对象集, 与其全部纯近邻共同形成一个簇; 否则转至 Step 3.

Step 3: 如果 D 非空, 则取出一个未经处理的点 p 并转至 Step 2; 否则转至 Step 4.

Step 4: 算法结束.

RDBClustering 算法的思想是先得到初始集, 再对初始集进行扩展进而得到聚类结果; RDBC_M 算法的思想是利用 DBSCAN 算法的思想, 结合相对密度的概念得到纯核心对象集, 然后经由纯核心对象集逐步扩展生成一个聚类.

2　增量式 RDBC_M 算法

2.1　影响集等相关概念

在聚类的生成过程中, 若在集合中插入或删除一个对象 p, 则可能会引起 p 纯近邻纯核心对象的地位发生变化, 从而导致一些对象之间的密度可达关系也随之变化. 所以, 必须讨论由于 p 的插入或删除影响到哪些对象, 同时还需将聚类变化的影响减少到最小范围, 避免对整个数据集进行搜索, 以加快聚类过程, 提高算法效率.

以下讨论均利用了纯近邻关系的如下特点: $N_{\mathrm{pure}}(p,k)$ 表示 p 的最近 k 个纯邻居, 但这种关系不满足对称性, 虽然 $q\in N_{\mathrm{pure}}(p,k)$, 但不能推断出 $p\in N_{\mathrm{pure}}(q,k)$, 即 p 有可能不是 q 的最近 k 个纯邻居之一. 为了便于后续讨论方便, 引入几个相关概念.

定义 9　对象 p 的父纯近邻邻居记为

$\text{Parent}N_{\text{pure}}(p,k)=\{q|\forall q\in D$ 使 $p\in N_{\text{pure}}(q,k)\}$.

类似于定义9, 将 $N_{\text{pure}}(p,k)$ 称为 p 的子纯近邻邻居.

定义 10 对于给定对象 p, 称

$$\text{Whole}N_{\text{pure}}(p,k)=\text{Parent}N_{\text{pure}}(p,k)\bigcup N_{\text{pure}}(p,k)\bigcup\{p\}$$

为 p 的全部纯近邻邻居.

在聚类的生成过程中, 若在集合中插入或删除一个对象 p, 则会引起 p 附近一些对象的纯邻域关系发生变化, 同时这些对象的绝对密度和其父纯近邻邻居的相对密度也会随之变化, 从而引起纯核心对象的地位改变. 如, 原来为纯核心对象, 因为对象 p 的插入或删除成为非纯核心对象.

由于增量聚类算法过程中可引起核心对象的地位改变, 本文采用类似文献 [7] 的思想, 提出种子更新对象和影响对象的概念.

定义 11 纯核心对象的地位因 p 的增删有可能发生变化的对象集合, 称为核心地位变化集, 记作 CorChangSet(p), 计算方法如下:

1) 插入时有

$$\text{CorChangSet}(p)=\text{Parent}\text{N}_{\text{pure}}(p,k)\bigcup\{p\}\bigcup\{q|\forall o\in\text{Parent}N_{\text{pure}}(p,k),\forall q\in\text{Parent}N_{\text{pure}}(o,k)\},$$

其中 Parent$N_{\text{pure}}(p,k)$ 为 p 插入后的父纯近邻邻居.

2) 删除时有

$$\text{CorChangSet}(p)=\text{Parent}N_{\text{pure}}(p,k)\bigcup\{q|\forall o\in\text{Parent}N_{\text{pure}}(p,k),\forall q\in\text{Parent}N_{\text{pure}}(o,k)\},$$

其中 Parent$N_{\text{pure}}(p,k)$ 为 p 删除后的父纯近邻邻居.

由于 CorChangSet(p) 是核心对象变化集, 集合 $D-$CorChangSet(p) 中的纯核心对象地位不会变化: 但 $\{q|q\in N_{\text{pure}}(o,k),o\in\text{CorChangSet}(p,k)\}$, 即核心地位变化集中对象的子纯近邻邻居的密度可达关系可能会发生变化. 如, 一个孤立点变成某个簇中的一个对象, 或者一个簇中的某个对象变成孤立点.

定义 12 给定阈值 $0<\lambda<1$, 若存在一个对象链 $p_1,p_2,\cdots,p_n$, 对于 $\forall p_i(i=1,2,\cdots,n)$ 有 $p_i\rightarrow^{\lambda}p_{i+1}$, 则称对象 p_n 从 p_1 出发关于阈值 λ 是相对密度可达的, 记为 $p_1\Rightarrow^{\lambda}p_n$.

定义 13 对象 p 在数据集 D 中的影响集是一个因 p 的插入或删除而使类标识可能改变的对象集, 定义为

$$\text{Affected}_D(p)=\text{CorChangSet}(p)\bigcup\{q|\exists o\in\text{CorChangSet}(p),\text{使得}o\Rightarrow q\}\bigcup\Big(\bigcup_{\forall q,\exists o\in\text{CorChangSet}(p)\wedge q\Rightarrow o}(\{q\}\bigcup N_{\text{pure}}(q,k))\Big).$$

直观地说, 影响集主要由核心地位变化集、所有从核心地位变化集中的对象出发且满足相对密度可达关系的对象、从这些对象出发与核心对象集中的任一对象满足相对密度可达关系的所有对象以及这些对象的纯子近邻邻居的集合等 3 部分构成. 因此, 为了减少聚类算法的搜索范围, 增量聚类算法只需关注影响集中的对象.

定义 14 设 p 为 D 中对象, 分别定义插入或删除 p 时的更新种子对象集为

$$\text{UpdSeeds}_{\text{Ins}}(p)=\{q|q\text{是}D\bigcup\{p\}\text{中的核心对象},\text{且}\exists o\text{CorChangSet}(p)\text{ 使 }q\in\text{Whole}N_{\text{pure}}(o,k)\},$$

$$\text{UpdSeeds}_{\text{Del}}(p)=\{q|q\text{是}D-\{p\}\text{中的纯核心对象},\text{且}\exists o\text{CorChangSet}(p)\text{ 使 }q\in\text{Whole}N_{\text{pure}}(o,k)\}.$$

由上述定义可知, 如果某个对象从非纯核心对象变成纯核心对象, 或从纯核心对象变成非纯核心对象, 则称为纯核心对象地位改变.

2.2 IncRDBC_M 算法

本文提出的基于相对密度的混合属性数据集增量聚类算法 IncRDBC_M, 其主要思想是根据对象的插入或删除分别进行不同的处理, 计算步骤如下.

Step 1: 从更新数据集中取出一个未处理的更新对象.

Step 2: 计算核心地位变化集、影响集和更新种子对象.

Step 3: 如果取出的点为插入对象, 则执行插入更新操作; 否则 (即取出的点为删除对象) 执行删除更新操作.

Step 4: 如果更新数据集非空, 则取出一个未处理的更新对象, 转至 Step 2; 否则转至 Step 5.

Step 5: 算法结束.

因为插入对象时, 有可能导致创建新簇、添加噪声、归入聚类或合并聚类等不同操作, 而删除对象时也可能导致消除聚类、删除噪声、移除簇内对象或者分裂聚类等不同操作, 所以, 算法的主要步骤是计算更新对象 p 的核心地位变化集 CorChangSet(p), 找出

其更新种子对象和影响集, 最后根据插入对象或删除对象的不同情形, 执行相应的聚类操作.

第 2.3 节和第 2.4 节分别介绍插入更新和删除更新的聚类操作算法.

2.3 插　入

当某个对象 p 插入到数据集中时, 可以出现 4 种不同情况:

1) 添加噪声.

① 若 $\text{UpdSeeds}_{\text{Ins}}(p)$ 为空, 则 $\text{CorChangSet}(p)$ 中没有引起纯核心地位改变的对象, 在 $\text{ParentN}_{\text{pure}}(o,k)$ 中也没有纯核心对象, 因此, 将 p 作为噪声标记, 其他对象保持不变.

② 若 $\text{UpdSeeds}_{\text{Ins}}(p)$ 非空, 且 $\text{CorChangSet}(p)$ 中存在因 p 的插入失去纯核心地位的对象 c, 则 $N_{\text{pure}}(c,k)\bigcup\{c\}\bigcup\{p\}$ 中的对象有可能变为孤立点, 判断方法是对 $N_{\text{pure}}(c,k)\bigcup\{c\}\bigcup\{p\}$ 中的对象 q 逐个进行是否为孤立点的判断, 若 $rd_{N_{\text{pure}}(q,k)}(q) < \lambda$, 且 $\text{Parent}N_{\text{pure}}(o,k)$ 中无纯核心对象, 则将 q 标为噪声.

2) 创建新簇.

$\text{UpdSeeds}_{\text{Ins}}(p)$ 非空, 如果该集中存在这样的新增的纯核心对象 o, 在 p 插入之前, o 不属于任何一个聚类, 且 p 插入后其相对密度可达对象中没有已知聚类中的纯核心对象, 则需要创建一个新的聚类. 由于新增的纯核心对象 o 可能有多个, 计算时首先将它们放入栈中, 然后依次从栈中取出并作为种子对象, 创建以 clusterID 为标识的新类, 再执行 Expand_Cluster(o, DB, k, λ, clusterID) 聚类扩展函数. 执行聚类扩展函数的主要原因是 $\text{UpdSeeds}_{\text{Ins}}(p)$ 中新增的纯核心对象可能不是唯一的.

3) 归入某一聚类.

① $\text{UpdSeeds}_{\text{Ins}}(p)$ 为空, 如果存在一个或多个纯核心对象 o (原先属于某个簇, 即不是噪声, 且对象 p 插入后, $N_{\text{pure}}(o,k)$ 中包含 p), 则将这些纯核心对象 o 放入临时对象集 O. 如果 O 中的对象原先仅属于一个簇, 则将 p 归入该簇; 如果 O 中的对象原先在不同簇中, 且不能合并, 则对于 O 中任何两个不同簇的对象, 因为它们彼此不能相对密度可达, 将 p 归入距离最近的那个簇.

② $\text{UpdSeeds}_{\text{Ins}}(p)$ 非空, 且其中仅有一个新增的纯核心对象 o, 若 $p \in N_{\text{pure}}(o,k)$ 且 o 原来就属于某个簇, 则将对象 p 归入 o 所在的那个簇.

③ $\text{UpdSeeds}_{\text{Ins}}(p)$ 非空且含有满足某种条件 A 的对象, 此外, $\text{Parent}N_{\text{pure}}(p,k)$ 中包含满足某种条件 B 的对象. 其中: 条件 A 指对象 o 原先不是纯核心对象, 现在成为新的纯核心对象, 且 $p \in N_{\text{pure}}(o,k)$; 条件 B 指对象 o 继续保持纯核心地位且 $p \in N_{\text{pure}}(o,k)$. 在这两种情况下, 对象 o 都要放入临时对象集 O 中.

若 O 中所有对象均在同一个簇中, 则将 p 归入该簇; 若 O 中对象原先不在同一个簇中, 且对应的簇不能合并, 则将 p 归入距离最近的那个簇, 否则将 p 归入合并的簇中.

4) 合并不同聚类.

若 $\text{UpdSeeds}_{\text{Ins}}(p)$ 非空且包含若干个新的纯核心对象 o, 则将 $\text{Whole}N_{\text{pure}}(o,k)$ (即 o 的全部纯近邻) 中的所有纯核心对象归入临时对象集 O 中. 若 O 中包括不止一个纯核心对象, 且它们原先不在同一个的簇中, 则需要完成如下操作: 对于 O 中任意两个属于不同簇的对象, 如果它们之间相对密度可达, 则将它们所在的两个簇合并成一个簇, 同时将原先的两个簇的类标识设置成相同. 该操作完成后的结果是, 或者 O 中所有对象在同一个簇中, 或者 O 中任意两个属于不同簇的对象之间均不能相对密度可达.

2.4 删　除

当从数据集中删除一个对象 p 时, 可能出现 4 种不同情况:

1) 删除噪声. 如果 p 是孤立点, 则将 p 删除后不会影响其他对象, 即其他对象的地位不变.

2) 消除一个聚类. 如果 $\text{UpdSeeds}_{\text{Del}}(p)$ 为空, 且删除前 p 不是孤立点, 删除 p 后核心地位变化集 $\text{CorChangSet}(p)$ 中没有任何纯核心对象, 则将包含 p 的簇中所有其他对象均标记为噪声.

3) 减少聚类中的对象. ① 若 $\text{UpdSeeds}_{\text{Del}}(p)$ 为空, 但 p 并非噪声, p 删除后, $\text{Parent}N_{\text{pure}}(p,k)$ 中仍包含纯核心对象, 则将 p 删除而余下的对象地位不变. ② 若 $\text{UpdSeeds}_{\text{Del}}(p)$ 非空, 则表明该集中的任意两个对象相对密度可达, 此时将 p 删除可能导致一些对象变为噪声, 因此需进行如下具体操作: 依次取出 $\text{CorChangSet}(p)$ 中纯核心对象地位发生变化的每个对象 o, 由于 o 及其纯近邻中的对象可能变成孤立点, 需要判断 o 和 $N_{\text{pure}}(o,k)$ 中的对象是否已变成孤立点, 并将其进行标记.

4) 分裂所在聚类. 若 $\text{UpdSeeds}_{\text{Del}}(p)$ 非空且有多个纯核心对象属于同一个簇, 其中还存在原先属于同一簇但相互不能直接相对密度可达的两个对象 a 和 b, 则需要重新判断是否可以通过邻域内的其他对象使 a 和 b 相对密度可达. 判断的计算可以限制在 a, b 的核心地位变化集 $\text{CorChangSet}(a)\bigcup\text{CorChangSet}(b)$ 上, 以便缩小计算搜索的范围, 避免遍历所有可能的对象.

若经过以上计算, 判断出 a 和 b 仍然不是相对密度可达的, 则将原来的簇分裂为两个或更多个簇, 分裂过程是分别以 a 和 b 作为聚类分裂的种子对象, 在此基础上依次进行扩展分裂.

3 实验与分析

3.1 参数选择

为了避免参数选择的盲目性和试探性, 本文借鉴文献[10]中参数选择的相关分析方法和思想, 推导出如下定理, 为参数 λ 的选择提供理论指导.

定理 1 设 D 是一个对象集, 用 min_dist 表示 D 中对象的最小 k 纯近邻距离, 即 min_dist = $\min\{k\_\mathrm{purdist}(p)|p\in D\}$. 类似地, 用 max_dist 表示 D 中对象的最大 k 纯近邻距离, 即 max_dist = $\max\{\mathrm{k\_purdist}(p)|p\in D\}$. 对于对象 $p\in D$, 如果 $N_{\mathrm{pure}}(p,k)\subseteq D$, 且 $\forall o\in N_{\mathrm{pure}}(p,k)$ 均有 $N_{\mathrm{pure}}(o,k)\subseteq D$, 则

$$0\leqslant rd_{N_{\mathrm{pure}}(p,k)}(p)\leqslant\frac{\max\_\mathrm{dist}}{\min\_\mathrm{dist}}-1.$$

证明 对于所有的 $p\in D$, 有 $k\_\mathrm{purdist}(p)\geqslant\min\_\mathrm{dist}$, 依照纯近邻密度的定义可知 $\mathrm{npnd}(p,k)\geqslant\min\_\mathrm{dist}$, 同理 $k\_\mathrm{purdist}(p)\leqslant\max\_\mathrm{dist}$, 因此 $\mathrm{npnd}(p,k)\leqslant\max\_\mathrm{dist}$. 依照上述讨论, 对于任意 $o\in N_{\mathrm{pure}}(p,k)$, 其纯近邻密度 $\mathrm{npnd}(p,k)$ 也在 min_dist 与 max_dist 之间, 因此按照相对密度的计算式(3), 若其求和式中分子与分母取值相同, 则 $rd_{N_{\mathrm{pure}}(p,k)}(p)$ 取得最小值 0; 若分子取值为 max_dist, 分母取值为 min_dist, 则 $rd_{N_{\mathrm{pure}}(p,k)}(p)$ 取得最大值 $\max\_\mathrm{dist}/\min\_\mathrm{dist}-1$, 故有

$$0\leqslant rd_{N_{\mathrm{pure}}(p,k)}(p)\leqslant\frac{\max\_\mathrm{dist}}{\min\_\mathrm{dist}}-1.\qquad\square$$

3.2 算法性能分析与实验

RDBC_M 算法是一个纯核心对象的判别和不断进行聚类扩展的过程. 对于给定的对象 p, 要判断它是否为核心对象, 就需对 p 进行一次 k 纯近邻的查询, 还要计算其相对密度, 而对象 p 的相对密度计算要使用 $N_{\mathrm{pure}}(p,k)$ 中每一个对象的 k 纯近邻绝对密度值 npnd, 即运行 RDBC_M 算法时一般需要多次读取对象的绝对密度值 npnd.

若在算法执行前对其进行预处理, 并将所有对象的绝对密度存储在中间数据表 Temp 中, 则可实现一次计算适应多次查询, 提高算法效率. 中间数据表 Temp 的维护任务可以通过数据库的触发器完成, 即当某个数据发生变更时, 由触发器自动计算该数据的绝对密度. 按照这种处理方法, 算法的时间复杂性便是查询 k 纯近邻的时间与查询中间数据表 Temp 的时间之和. 由于查询 k 近邻的时间复杂性为 $O(n\log n)$, 查询中间数据表 Temp 的时间复杂性为 $O(n)$, 求和可得算法的时间复杂度为 $O(n\log n)$, 与算法 DBSCAN 的时间复杂性同阶.

下面进行两组实验, 并在实验数据的基础上, 结合理论分析比较聚类算法 RDBC_M 和增量式聚类算法 IncRDBC_M 的性能差异.

实验数据 1 是二维空间中的模拟数据, 如图 1 所示. 图 1 中, 横坐标与纵坐标的数值表示数据对象在二维空间的位置信息, 单位均为 cm. 数据有 3 个属性, 除了 2 个位置属性外, 还有 1 个分类属性代表形状; 分为 3 个簇, 每个簇的密度不同. 在第 1 个簇中, 每个对象与其上下、左右对象的距离均为 1 个单位, 在第 2 个簇和第 3 个簇中, 每个对象与上下、左右对象的距离分别为 2 和 3 个单位. 数据生成的基本过程如下: 1) 生成数据坐标为 (x,y) 的矩形状簇; 2) 用 $(x+0.1\times d\times\mathrm{rand}(\ ),y+d\times\mathrm{rand}(\ ))$ 修改 (x,y) 的坐标位置, 其中 d 为簇中一个对象上下、左右对象之间的距离, rand( ) 为一个随机函数, 它生成 [0,1] 之间的随机数; 3) 在矩形簇中删除一些连续相邻的点, 这样可以使簇的形状不再是一个正矩形. 实验数据 2 为三维空间中的模拟数据, 如图 2 所示. 图 2 中, x,y,z 坐标数值为数据对象在三维空间的位置信息, 单位均为 cm, 且一个簇包裹着另外一个簇, 数据的生成方法类似于实验数据 1 的生成.

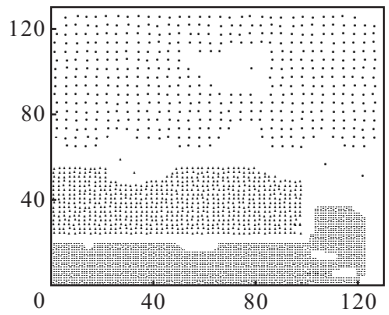

图 1 二维空间实验数据

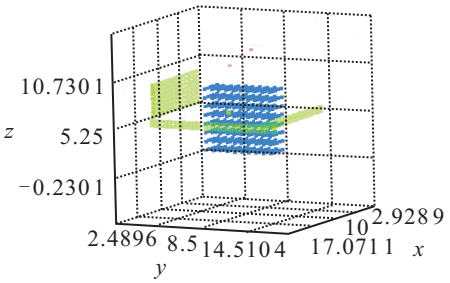

图 2 三维空间实验数据

参数 k 取为 4～7, 相对密度的阈值 λ 设定为 0.2, 若两个对象的分类属性值相同, 则将其距离设置为 0, 否则, 将其距离设置为 ∞ . 经过多次仿真实验, 均能得到相同的聚类结果.

RDBC_M 算法与 DBSCAN 算法类似, 均以某个核心对象作为种子对象进行聚类扩展. RDBC_M 算法每次聚类扩展都要进行一次区域查询, 即查询种子对象的 k 近邻邻居. 若将 RDBC_M 算法的时间复杂性记为 $\mathrm{Cost}_{\mathrm{RDBC\_}M}(n)$, 则对于 n 个对象的数据集, 其算法最多执行 n 次区域查询, 因此有 $\mathrm{Cost}_{\mathrm{RDBC\_}M}(n)=n$.

增量式 IncRDBC_M 算法需要执行的区域查询次数完全由实际情况而定. 如, 对象的更新数量、删除对象数量和插入对象数量的比值变化等, 都会引起区域查询次数的变化. 表 1 给出一些性能参数以方便表达算法的性能开销. 若要更新 $\mathrm{num_{upd}}$ 个对象, 可将 IncRDBC_M 算法执行的时间复杂性记为

$$\mathrm{Cost}_{M\_\mathrm{IncRDBC}}(\mathrm{num_{upd}})=\mathrm{num_{upd}}(\mathrm{per_{ins}}\times\mathrm{reg_{ins}}+\mathrm{per_{del}}\times\mathrm{reg_{del}}). \quad (5)$$

表 1　性能分析的参数

参数	含　义
n	数据集中数据对象的个数
$\mathrm{num_{upd}}$	对象更新的个数, 是插入和删除对象的个数之和
$\mathrm{per_{ins}}$	对象插入数与对象更新数的比例
$\mathrm{per_{del}}$	对象删除数与对象更新数的比例
$\mathrm{reg_{ins}}$	一个对象插入时需要执行区域查询的平均次数
$\mathrm{reg_{del}}$	一个对象删除时需要执行区域查询的平均次数

表 2 为执行 IncRDBC_M 算法获取的实验数据.

表 2　实验数据

参数	二维数据集	三维数据集
n	5 000	727
$\mathrm{num_{upd}}$	动态变化	动态变化
$\mathrm{per_{ins}}$	0.5	1
$\mathrm{per_{del}}$	0.5	0
$\mathrm{reg_{ins}}$	1.8	1.6
$\mathrm{reg_{del}}$	7.2	0

下面定义增量聚类算法 IncRDBC_M 相对于原非增量聚类算法的性能“加速因数”, 它是增量聚类算法与原非增量聚类算法的性能开销的比值, 即

$$\mathrm{SpeedupFactor}=\frac{\mathrm{Cost}_{M\_\mathrm{RDBCA}}(n+\mathrm{num_{upd}}(\mathrm{per_{ins}}-\mathrm{per_{del}}))}{\mathrm{Cost}_{M\_\mathrm{IncRDBCA}}(\mathrm{num_{upd}})}=\frac{n+\mathrm{per_{ins}}\times\mathrm{num_{upd}}-\mathrm{per_{del}}\times\mathrm{num_{upd}}}{\mathrm{num_{upd}}\times(\mathrm{per_{ins}}\times\mathrm{reg_{ins}}+\mathrm{per_{del}}\times\mathrm{reg_{del}})}. \quad (6)$$

因此, 只要将仿真实验计算得到的数据代入式 (6) 即可得出加速因数的值. 在数据总量为 n 的数据集上, 对 $\mathrm{num_{upd}}$ 个对象进行更新的两组实验加速因数变化情况如图 3 和图 4 所示.

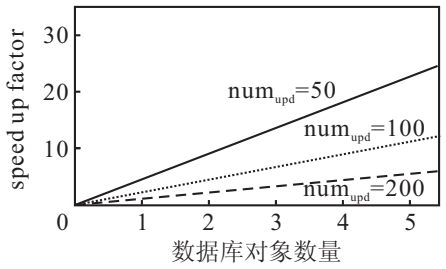

图 3　二维数据情形的加速因数

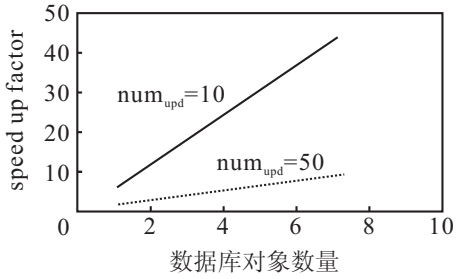

图 4　三维数据情形的加速因数

从图 3 和图 4 可以看出, 执行增量聚类算法时, 数据对象更新得越少, 其加速因数的增长越明显.

3.3　聚类质量

本文提出的聚类算法 RDBC_M 和增量聚类算法 IncRDBC_M 具有以下特点: 1) 对混合属性数据集适用; 2) 抗噪能力较强; 3) 可以发现任意形状的簇; 4) 克服了参数敏感与选择问题; 5) 能够区分不同密度等级的簇. 对于图 1 和图 2 所示的数据集, 能够轻易地得出正确的聚类结果, 而 DBSCAN 在参数选取不当时可能得不到正确的聚类结果.

相对于 RDCTD, RDBClustering, RDBIClustering 等聚类算法, 本文提出的聚类算法具有以下几点不同和改进: 1) 定义了混合属性距离、纯核心对象等概念, 对混合属性的数据集适用, 传统的 RDBClustering 算法和 DBSCAN 算法均没有考虑非数值属性, 但空间数据库中不仅保存了空间物体的数值属性, 也保存了其非数值属性 (如分类属性), 当分类属性在聚类中起很大作用时, 这些传统聚类算法便无法使用; 2) 为了更准确地反映同一个簇中对象间密度的平均差异值, 改进和优化了相对密度计算公式; 3) 提出了纯核心对象集的概念, 并将簇和纯核心对象集统一看待, 保证簇中对象形成一个紧密的整体.

本文较文献 [10] 给出的基于相对密度的增量聚类算法描述得更具体和详尽, 充分考虑了数据插入更新和删除更新时出现的各种聚类变化和操作方法, 而文献 [10] 方法不仅未对删除更新进行描述, 且未将聚类变化的各种情况考虑完全.

4 结 论

本文提出的基于相对密度的混合属性数据集增量聚类算法IncRDBC_M, 不仅保持了传统密度聚类算法对噪声数据不敏感和可以发现任意形状簇的优点, 还具有不同密度等级的聚类能力. 算法在聚类时同时考虑数值属性和非数值属性, 利用纯邻居、相对密度、纯核心对象集等概念, 使每个簇中的对象能够更加紧密地形成一个整体. 另外, 提出的聚类算法在参数设置方面具有可供参考的理论依据, 避免了参数选择的盲目性和随意性.

IncRDBC_M以单个对象的插入或删除而触发执行, 本文首先定义了核心地位变化集, 进而确定了增删对象的影响集, 最后分别讨论了在该影响集上插入或删除某单个对象时对聚类的影响, 并通过实验验证了增量式聚类算法的有效性和性能加速效果. 与增量式DBSCAN算法相比的主要优点是, IncRDBC_M可以找到密度等级不同的聚类结果并适用于混合属性数据集.

参考文献(References)

[1] 孙焕良, 邱菲, 刘俊岭, 等. IncSNN——一种基于密度的增量聚类算法[J]. 计算机研究与发展, 2006, 43(3): 309-313.
(Sun H L, Qiu F, Liu J L, et al. IncSNN: An incremental clustering algorithm based on density[J]. J of Computer Research and Development, 2006, 43(3): 309-313.)

[2] Ester M, Kriegel H P, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise[C]. Proc of the 2nd Int Conf on Knowledge Discovery and Data Mining. Portland: AAAI Press, 1996: 226-231.

[3] 李小畅. 一种改进的基于相对密度的聚类算法[C]. 2011年中国信息技术应用学术研讨会论文集. 北京: 电子工业出版社, 2011: 29-38.
(Li X C. An improved relative density-based spatial clustering algorithm with noise[C]. Proc of 2011 China Symposium on Information Technology Application. Beijing: Electronic Industry Press, 2011: 29-38.)

[4] 刘青宝, 邓苏, 张维明. 基于相对密度的聚类算法[J]. 计算机科学, 2007, 34(2): 192-195.
(Liu Q B, Deng S, Zhang W M. Relative density-based clustering algorithm[J]. Computer Science, 2007, 34(2): 192-195.)

[5] Ajaya K Akasapu, Srinivasa Rao P, Sharma L K, et al. Density based k-nearest neighbors clustering algorithm for trajectory data[J]. Int J of Advanced Science and Technology, 2011, 31: 47-58.

[6] Chen Jin-hua, Chen Xiao-yun. Relative density weights based fuzzy C-means clustering algorithms[J]. Quantitative Logic and Soft Computeing, 2010, 82(2): 459-466.

[7] Ester M, Kriegel H P, Sander J, et al. Incremental clustering for mining in a data warehousing environment[C]. Proc of the 24th Int Conf on Very Large Data Base. New York: Morgan kaufmann Publishers Inc, 1998: 323-333.

[8] 黄永平, 邹力鹃. 数据仓库中基于密度的批量增量聚类算法[J]. 计算机工程与应用, 2004, 40(29): 206-209.
(Huang Y P, Zou L K. An incremental density-based clustering algorithm in a batch mode used in a data warehouse[J]. Computer Engineering and Applications, 2004, 40(29): 206-209.)

[9] 徐新华, 谢永红. 增量聚类综述及增量DBSCAN聚类算法研究[J]. 华北航天工业学院学报, 2006, 16(2): 15-17.
(Xu X H, Xie Y H. Summarization on incremental clustering and research of incremental DBSCAN algorithm[J]. J of North China Institute of Astronautic Engineering, 2006, 16(2): 15-17.)

[10] 刘青宝, 侯东风, 邓苏, 等. 基于相对密度的增量式聚类算法[J]. 国防科技大学学报, 2006, 28(5): 73-79.
(Liu Q B, Hou D F, Deng S, et al. Relative density based incremental clustering algorithm[J]. J of National University of Defense Technology, 2006, 28(5): 73-79.)

[11] Woo K G, Lee J H. FINDIT: A fast and intelligent subspace clustering algorithm using dimension voting[D]. Taejon: Korea Advanced Institute of Science and Technology, 2002.

[12] Breunig M M, Kriegel Hans-peter, Ng R T, et a1. Identifying density-based local outliers[C]. Proc ACM Sigmod Int Conf Manage Data. New York, 2000: 93-104.

[13] 黄德才, 吴天虹. 基于密度的混合属性数据流聚类算法[J]. 控制与决策, 2010, 25(3): 416-421.
(Huang D C, Wu T H. Density-based clustering algorithm for mixture data sets[J]. Control and Decision, 2010, 25(3): 416-421.)