数据挖掘中基于密度和距离聚类算法设计

田 地，王世卿
（郑州大学 信息工程学院，河南 郑州 450052）

摘 要：介绍聚类分析的基本概念，并说明了关于聚类分析相关研究工作。对聚类、数据对象、对象的密度、簇的密度、距离和 ε－邻域等基本概念进行了描述。在此基础上提出并分析了基于密度和距离聚类算法，并与其他聚类方法作了比较，显示了其优越性。

关键词：数据挖掘；聚类；距离；对象的密度
中图分类号：TP301.6 **文献标识码**：A **文章编号**：1673－629X(2006)10－0049－03

Design of Density and Distance－Based Clustering Algorithm in Data Mining

TIAN Di, WANG Shi-qing
(Sch. of Information Engineering , Zhengzhou University, Zhengzhou 450052, China)

Abstract: Presents the basic concept of clustering and its correlating research work. Following the description of the concepts, such as clustering, data object, density of object, density of cluster, distance and ε－neighbor, the algorithm of density and distance based clustering are introduced and analyzed. The algorithm shows its superiority after compared with other methods of clustering.

Key words: data mining; clustering; distance; density of object

0 引 言

数据挖掘能自动发现隐藏在数据库、数据仓库或海量信息存储中的知识模式，因此数据挖掘又称作数据库中的知识发现。聚类分析是数据挖掘的重要的组成部分，它是将物理或抽象对象的集合分组成由类似对象组成的多个类的过程，它的目标是使在同一类中的数据的相似性较其它类更近。目前，它已成为数据挖掘研究领域中一个非常活跃的研究方向。

聚类分析技术在模式识别、数据分析、图像处理和市场研究等许多领域得到了广泛的应用，例如，在数据分析领域，通过比较数据的相似性和差异性，能发现数据的内在特征和发布规律，从而获得对数据更深刻的理解和认识；在模式识别领域，利用聚类技术对一些过程或时间进行分类等。

聚类分析属于一种无监督的学习方法，它是一种通过观察学习的方法，不是示例学习方法。聚类分析的一些应用对分析算法提出了特别的要求，下面列出一些典型的要求：

(1)可伸缩性：指算法要能够处理大数据量的数据库对象。当算法不能处理大数据量时，也不提倡用抽样的方法，因为这通常导致歪曲的结果。

(2)处理不同数据类型：算法不仅要能处理数值性的字段，还要有处理其它类型字段的能力，例如：布尔型、序数型、枚举型等。

(3)发现具有任意形状的聚类：很多聚类分析算法采用基于欧几里得距离的相似性度量方法，这一类算法发现的聚类通常是一些球状的、大小和密度相近的类；但现实数据库中的聚类可以是任意形状的，故要求算法有发现任意形状聚类的能力。

(4)输入参数对领域知识的弱依赖性：很多聚类算法都要求用户输入一些参数，聚类分析的结果对这些参数是很敏感的；另一方面，对于高维数据，这些参数又是很难确定的。这样就加重了用户使用这个工具的负担，使得分析的结果很难控制。

(5)其他能力：如能够处理异常数据，结果对数据的输入顺序具有无关性，处理高维数据的能力，结果的可解释性和可用性。

1 相关的工作的综述

近年来，数据库研究人员根据数据库中数据量大的特点，提出了新的聚类算法。

当前的聚类方法可以分为以下5类。

1.1 基于划分的聚类方法

其基本思想是根据假定的数据分布，将数据分为 K 个组，利用循环再定位技术（通过移动不同划分组中的对象）来改变划分的内容，并且这种划分使得所用的划分评

收稿日期：2005－12－29
作者简介：田 地(1980－)，男，河南项城人，硕士研究生，研究方向为人工智能、信息系统；王世卿，教授，研究方向为系统控制、电子政务。

判标准达到最优,事实上就是用中心、平均、中值扩散到难于确定的点,并用一个点代表一个类。这一类启发聚类算法在分析中小规模数据集发现圆形或球形聚类时工作的比较好,但它不能识别任意形状的聚类。基于划分的方法主要有 K-means[1],CLARANS[2]等算法。

1.2 基于分层的聚类方法

基于分层的方法主要有 BIRCH[3],CURE[4]等,其基本思想是对给定的数据对象集合进行层次分解(自底向上或自顶向下),自顶向下的观点认为聚类是将不同种类的数据分割为同一种类;自底向上的观点认为聚类是在数据集中根据一些自然相似的标准发现组。分层聚类算法的时间复杂度为 $O(n^2)$。

1.3 基于密度的聚类方法

基于密度的聚类的算法有 DBSCAN[5],OPTICS[6]等。其主要思想是只要临近区域的密度超过给定的阈值,就继续聚类,即对给定类中的每个点,在一个给定范围的区域中至少包含某个数目的点,这样的方法可以用来过滤噪声/孤立点,发现任意形状的聚类。但是 DBSCAN 对用户的参数是非常敏感的,算法中使用全局密度参数来刻画内在聚类,所以不适用于带有不同密度的聚类。为了解决数据分布不均匀的聚类问题,提出了 OPTICS 算法,它是对不同密度的对象进行排序,以便按特定的顺序处理。这两个算法的时间复杂度都是 $O(n\log n)$(采用空间索引时);DENCLUE 用影响函数来刻画每个点的影响,数据空间的整体密度可以模型化为所有点的影响函数的总和,然后通过确定密度吸引点聚类。它适用于高维空间的聚类,有坚实的数学基础和良好的聚类结果,对噪声不敏感。但是对密度参数和噪声阈值的选择可能影响聚类的质量。

1.4 基于网格的聚类方法

基于网格的聚类主要有 STING[7], CLIQUE[8], WAVE-CLUSTER[6]等,这一类方法都是把对象空间划分为有限数目个的单元,形成网格结构,所有的聚类都是在网格上进行的。主要优点是处理速度快,其处理时间独立于对象的数目,仅依赖于量化空间中每一维上的单元数。STING 是一种基于网格的多分辨率聚类技术,它将空间区域划分为矩形单元,针对不同的分辨率,通常存在多个级别的矩形单元,这些单元形成一层次结构高层的单元被划分为多个低一层的单元。算法的时间复杂度为 $O(n)$。该算法的聚类的质量取决于网格结构最低层的粒度,尽管该技术有快速的处理速度,但可能降低聚类的质量和精确性。WAVE-CLUSTER 是一个基于网格和密度的算法,它首先通过在数据空间上强加一个多维网格结构来汇总数据,然后采用小波变换来变换原有的特征空间,在变换的空间中找到密集区域。该算法提供了无指导的聚类,速度快,时间复杂度为 $O(n)$,能有效地处理大数据集合,发现任意形状的聚类,成功地处理孤立点,对输入的顺序不敏感,聚类的质量好、效率高。CLIQUE 也是基于网格和密度的算法,它是根据下面的一个事实开发出的算法:如果一个 K 维单元是密集的,那么它在 $K-1$ 维空间的投影也是密集的。该算法对元组的输入顺序不敏感,具有良好的可伸缩性。

1.5 基于模型的聚类方法

基于模型的聚类方法为每个簇假定了一个模型,寻找数据对给定模型的最佳拟合。它又分为两类:统计学方法和神经网络方法。

2 文中的工作

提出了一种基于密度和距离聚类算法 DDBC(density and distance based clustering)。DDBC 算法受 K-means 算法和 DBSCAN 算法的启发。与 K-means 算法不同的是,簇的个数并不预先确定;与 DBSCAN 算法不同的是,每个数据对象只要满足距离阈值,就可加入到类中。DDBC 算法采用密度的方法屏蔽异常数据(噪声)对算法的影响,与 DBSCAN 算法在处理任意形状聚类方面的能力接近。由于距离的方法可以将对高维数据的处理转换到一维空间[9],在这个意义上,DDBC 可以处理高维数据。多维空间与二维空间的距离计算相似,为了方便描述算法,文中以二维空间为例来分析 DDBC 算法。

3 DDBC 聚类算法

3.1 基本概念

定义 1 数据对象的聚合称作簇(cluster)。

其中聚合是指由两个或更多个数据对象所构成的有目的的集合。聚合形象地说明了簇的特征,即同一簇中的数据对象相似,不同簇中的数据对象相异。

定义 2 把一个数据对象集合分成多个簇称作聚类分析(cluster analysis)。

把这些簇分别记作 $C_1, C_2, \cdots, C_n$,则应有

$C_i \cap C_j = \varnothing \ (i \neq j, i < j, j \leqslant n)$

定义 3 类分析的过程称作聚类(clustering)。

定理 1 聚类结果所形成的簇的数目不小于 2。

证明:反设聚类所形成簇的数目等于 1,与定义 2 矛盾。得证。

定义 4 以数据对象 o 为圆心,半径 ε 内的区域称 o 的 ε-邻域(ε-neighborhood)。

定义 5 设类 C 是一个聚类,α 是一个数据点,α 到聚类 C 的距离定义为 α 到 C 中各个点的最短距离,即 $d(\alpha, C) = \min\{d(\alpha, c) \mid c \in C\}$,其中 $d(\alpha, c)$ 表示两个点间的距离,$|C|$ 表示聚类 C 中的元素个数。

定义 6 对象 α 的 ε-邻域内的数据个数称为对象 α 的密度,用 $d(\alpha)$ 表示。

定义 7 若 $d(\alpha, \beta) < \varepsilon$,$\alpha \in$ 聚类 C,则称 β 按 ε 属于聚类 C。

定理 2 C_1, C_2 是两个聚类,α 是一个数据点,$d(\alpha, C_1) < \varepsilon$,且 $d(\alpha, C_2) < \varepsilon$,则 $C_1 = C_1 \cup C_2 \cup \{\alpha\}$。

证明:由于 $d(\alpha, C_1) < \varepsilon$,所以根据定义 7,$\alpha$ 应该按

ε 属于聚类 C_1;同理,α 应该按 ε 属于聚类 C_2,故说明 C_1, C_2 属于同一个聚类。结论成立。

定义 8　对于给定的 MinPts 和聚类 C,若 $|C| <$ MinPts,则称 C 是基于距离的噪声类;若对象的 ε - 邻域的密度小于给定的阈值 MinDPts,则称该对象是基于密度的噪声。

3.2　相应的算法

输入:阈值 ε,阈值 MinPts,数据集 D,密度阈值 MinDPts

输出:聚类,噪声

```
ClusteringI(D,ε,MinPts,MinDPts)//|D| = n
{
  select p∈D,create(C),assign(p,C),set(C.label,1)
  clustercount = 1
  for i = 1 to n do
  {
  clustercountDynamic = clustercount
  add = 0;Clabel = 0
  for j = 1 to clustercount do
    {
    if d(α, Cj) < ε and add = 0 then
    assign(α, Cj);Clabel = j;add = 1;
    else if d(α, Cj) < εand add <>0 then
      {merge(Cclabel,Cj);delete(Cj); clustercountDynamic
- ;
      }
      else
      {create(C);set(C.label, + + clustercountDynamic);as-
sign(α,CclustercountDynamic);
      }
    }
  sort(every C)//对聚类按其对象的个数排序(升序)
  clustercount = clustercountDynamic
  }
for(every C)
    {
    if |C| < MinPts then output(C)
    if α∈C and α.count < MinDPts then output(α)
    }
}
```

算法过程解释:首先是随机选择一个点作为第一个类,并将类标号设置为 1;已聚类的总数为 1;两个嵌套的 FOR 循环是对数据集中的每个点,检查它的 ε - 邻域是否包含已经形成的类中的点。其中 add 是判断被检查的点是否已加入到某个聚类中,Clable 标记被检查的点加入了哪个聚类中;内层的 FOR 循环构成算法的核心,若 α 属于某个聚类,就将它加入到该聚类中,并记录该聚类的标号,修改加入标志 add;若 α 属于某个聚类,但该点已经加入到前面的聚类中了,就将两个聚类合并,并删除该聚类,并修改聚类总数;若该点不属于任何的聚类,就创建一个聚类,给它一个标号;计算 $d(\alpha, C_i)$ 事实上是计算 α 同 C_i 中每个点的距离的过程,在计算 $d(\alpha,\beta)$ 的同时,若 $d(\alpha,\beta) < \varepsilon$,则 α,β 的密度都要加 1。对一个点处理后,要对聚类按其极点的个数排序(升序),并重新给每个聚类一个标号,由于上面记录了每个点的密度,所以第 15~17 行是输出基于距离的噪声点和基于密度的噪声点。事实上基于距离的噪声点是全局噪声,基于密度的噪声点是局部噪声。

在算法中,距离使用欧几里得距离,即:

$$d(\alpha,\beta) = \sqrt{|x_1 - y_1|^2 + |x_2 - y_2|^2 + \cdots + |x_p - y_p|^2}$$

这里,$\alpha = (x_1, x_2, \cdots, x_p)$,$\beta = (y_1, y_2, \cdots, y_p)$。

3.3　输入参数的设定

DDBC 算法有 4 个参数:数据集 D,距离阈值 ε,阈值 MinPts,密度阈值 MinDPts。其中 MinPts 控制基于距离的噪声点的输出,MinDPts 控制基于密度的噪声点的输出。

3.4　算法性能分析

使用的测试数据集来自 DBSCAN 算法中的 database (DS),在 VC6.0 环境下测试通过了该程序。DDBC 算法设计简单,保持了基于密度算法可以发现任意形状的聚类和对噪声数据不敏感的优点,空间复杂度小($O(n)$),时间复杂度为 $O(n^2)$,能够发现局部噪声和全局噪声。但对选定的阈值是敏感的,如果 ε 选择的不合适,它会成为两个聚类间的桥梁,从而把两个类看成一个类,不适合多密度的数据分布。

4　结束语

采用距离的方法可以将对高维数据的处理转换到一维空间,在这个意义上,DDBC 可以处理高维数据。下一步的工作是测试 DDBC 算法在高维情况下的性能。

参考文献:

[1] Kaufan L, Rousseeuw P J. Finding Groups in Data:an Introduction to Cluster Analysis[M]. New York: John Wiley & Sons, 1990.

[2] Ng R T, Han J. Efficient and effective clustering methods for spatial data mining[A]. In:Bocca J B,Jake M,Zaniolo C. Proceedings of the 20th VLDB Conference[C]. San Francisco: Morgan Kaufmann,1994. 144 - 155.

[3] Zhang T, Ramakrishnan R, Livny M. BIRCH: an efficient data clustering method for very large databases[A]. In: Jagadish H V, Munlick I S. Proceeding of the ACM SIGMOD International Conference on Management of Data[C]. New York: ACM Press,1996. 103 - 114.

[4] Guha S, Rastogi R, Shin K. CURE: an efficient clustering algorithm for large databases[A]. In:Haas L M, Tiwary A. Proceeding of the ACM SIGMOD International Conference on Management of Data[C]. New York: ACM Press, 1998. 73 - 84.

(下转第 54 页)

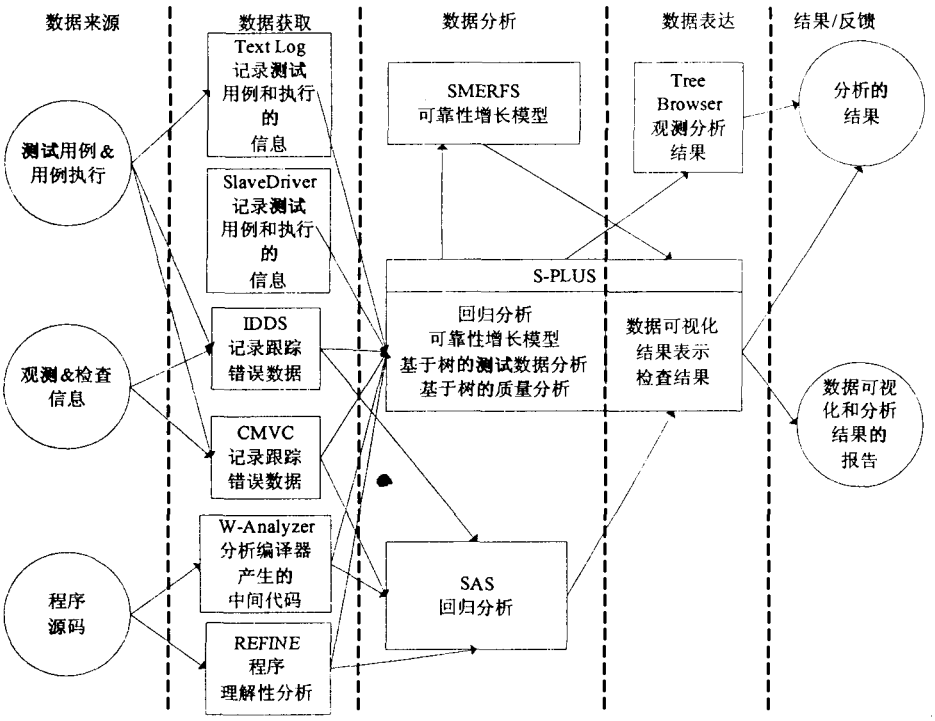

图 2 质量测量、分析和反馈工具

* 数据分析工具:用于帮助人们执行数据处理、分析和建模。

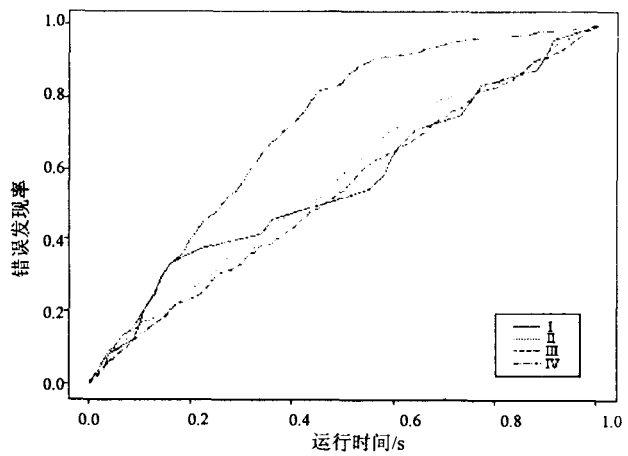

图 3 测试效率对比

* 数据表达工具:用于帮助人们检查分析的结果和提出适当的提高软件质量的方法。

实际中采用这种测试流程,使用相关的测试工具,软件测试会达到显著的效果。图 3 中,在软件设计中使用 4 种不同的测试方法(其中 IV 代表文中的测试方法),从 4 条不同的曲线可以看出文中所提出的方法效率较好。

3 结 论

由于软件的规模越来越大,软件测试的重要性更加突出,软件测试效率的提高对于软件测试的质量有着至关重要的作用。而软件测试效率主要从测试前期的计划、测试目标的选择、测试事例的设计和测试数据的测量和评估等几个方面来提高。采用文中提出的测试流程,在实际运用中可以提高软件测试的效率,保证软件质量。

参考文献:

[1] Tian J. Software Quality Engineering Testing, Quality Assurance, and Quantifiable Improvement [M]. USA: Addison Wesley, 2005.

[2] ISO(2001). ISO/IEC 9126 - 1:2001 Software Engineering - Product Quality - Part I:Quality Model. ISO[S].2001.

[3] Marick B. Classic Testing Mistakes[EB/OL]. http://www.testing.com/writings.html, 1997.

[4] Dustin E. Effective software testing - 50 specific ways to improve your testing[M]. USA:Addison Wesley, 2002.

[5] 赵 彬,辛文逵.目前软件测试发展中的误区[J].信息与电子工程,2003,12(4):323 - 325.

(上接第 51 页)

[5] Ester M, Kriegel H P, Sander J, et al. A density - based algorithm for discovering cluster in large spatial databases with noise[A]. In: Simoudis E, Han J, Fayyad U. Proceedins of the 2nd International Conference Knowledge Discovering in Databases and Data Mining (KDD - 96)[C]. Massachusetts: AAAI Press, 1996. 226 - 232.

[6] HAN Jiawei, KAMBER M. Data Mining, Concepts and Techniques[M]. CA: Morgan Kaufmann Publishers, 2000.

[7] Zhang W, Yang J, Muntz R. STING: a statistical information grid approach to spatial data mining[A]. In: Jarke M, Careg M J, Dittrich K R, et al. Proceedings of 23rd VLDB Conference[C]. San Francisco: Morgan Kaufmann, 1997. 186 - 195.

[8] Agrawal R, Gehrke J, Gunopulos D, et al. Automatic subspace clustering of high dimensional data for data mining applications [A]. In: Haas L M, Tiwary A. Proceedings of the ACM SIGMOD International Conference on Management of Data[C]. New York: ACM Press, 1998. 94 - 105.

[9] Berchtold S, Hohm C, Kriegel H P. The pvramid - technique: towards breaking the curse of dimensionality[A]. In: Haas L M, Tiwary A. Proceedings of the ACM SIGMOD International Conference on Management of Data[C]. Scattle: ACM Press, 1998. 142 - 153.