

文章编号:1001-9081(2005)02-0286-03

一种基于引力的聚类方法

蒋盛益^{1,2}, 李庆华²

(1. 衡阳师范学院 计算机系, 湖南 衡阳 421008;

2. 华中科技大学 计算机科学与技术学院, 湖北 武汉 430074)

(jiangshengyi@163.com)

摘要:将万有引力的思想引入聚类分析中,提出了一种基于引力的聚类方法 GCA (Gravity-based Clustering Approach),同时给出了一种计算聚类阈值的简单而有效的方法。GCA 关于数据库的大小和属性个数具有近似线性时间复杂度,这使得聚类方法 GCA 具有好的扩展性。实验结果表明 GCA 可产生高质量的聚类结果。

关键词:引力;聚类;阈值;数据挖掘

中图分类号: TP311.12 **文献标识码:** A

Gravity-based clustering approach

JIANG Sheng-yi^{1,2}, LI Qing-hua²

(1. Department of Computer Science, Hengyang Normal University, Hengyang Hunan 421008, China;

2. School of Computer Science and Technology, Huazhong University of Science & Technology, Wuhan Hubei 430074, China)

Abstract: The idea of the law of gravity was introduced to clustering analysis, and a gravity-based clustering approach, named GCA, was presented in this paper. At the same time, a simple method to calculate cluster threshold was put forward. The clustering approach GCA had the nearly linear time complexity with the size of dataset and the number of attributes, which resulted in good expandability. The experimental results show that the GCA creates high quality clustering.

Key words: Gravity; Clustering; Threshold; data mining

0 引言

所谓聚类是将物理或抽象的集合分成相似对象组成的多个类的过程,使得同一类中对象间的相似度最大化,而不同类中对象间的相似度最小化。聚类分析是数据挖掘的一个非常活跃的研究分支,具有广泛的应用。典型应用包括生物学上的基因分类和动植物分类,商务上的对客户群体的分类,科学研究数据的探索,信息检索与文本挖掘,通过对 Internet 上 Web 文档的聚类分析来发掘有用信息。聚类分析可以作为一个独立的工具来使用,帮助获取数据分布情况,了解各数据类的特征,确定所感兴趣的数据类以做进一步的分析;也可以作为其他算法(如特征构造与分类等)的预处理步骤,在聚类分析所生成的类上进一步处理。目前已有许多聚类算法^[1-7],但绝大部分不能有效处理混合属性的数据集,有些方法只能用于分类属性的数据集^[1-3],而有些只能用于数值属性的数据集^[4-5]。尽管文献[6]可以用于混合属性数据集,但对分类属性处理非常繁琐,且效果不理想。文献[7]综合了这些聚类算法的优点,提出了适合于混合属性数据集的聚类算法。已有的聚类方法存在以下不足:1)聚类结果对阈值较敏感,没有确定阈值的一般原则。2)在聚类过程中只考虑元素与类间距离,而没有考虑类大小的作用。如图1所示,设A1包含200个对象,A2包含140个对象,对象p到两个类的质心间的距离相等,则p应放到A1中,而不是随便放到哪一个中,甚

至即使与A1间的距离稍大,也要放入A1中。

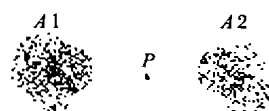


图1 类大小对聚类的影响

本文针对上述不足,将万有引力的思想引入聚类分析中,提出了一种经过一趟扫描就能得到聚类结果的有效聚类方法 GCA (Gravity-based Clustering Approach)。理论分析表明,聚类方法 GCA 具有近似线性时间复杂度和好的扩展性,可应用于大规模数据集的聚类分析。基于最小距离原则的聚类方法有可能将大的自然聚类划分成了小的聚类,聚类方法 GCA 并不限制聚类的半径,按照引力将数据划分为半径不同的超球体,可更好地保持自然聚类,减少聚类个数,提高聚类质量,实验结果有力地支持了这一想法。

1 基本概念

我们给出一组定义,将 Mahattan 距离概念推广到一般的数据空间。假设数据集 D 有 m_c 个分类属性和 m_n 个数值属性, $m = m_c + m_n$,不妨设分类属性位于数值属性之前,用 D_i 表示第 i 个属性及其取值的集合,由于对象与其标识(可理解为记录号)是唯一对应的,有时也就将一个对象与其标识等同起来。

收稿日期:2004-07-13;修订日期:2004-10-12 基金项目:国家自然科学基金资助项目(60273075)

作者简介:蒋盛益(1963-),男,湖南隆回人,副教授,博士研究生,主要研究方向:数据挖掘和并行算法;李庆华(1940-),男,湖北武汉人,教授,博士生导师,主要研究方向:并行算法、网格计算、数据挖掘等。

定义1 给定类 $C, a_i \in D_i, a_i$ 在 C 中关于 D_i 的支持度定义为 C 在 D_i 上的投影中包含 a_i 的次数:

$$Sup_{C \mid D_i}(a_i) = |\{object \mid object \in C, object.D_i = a_i\}|.$$

定义2 给定类 C, C 的摘要信息 $CSI(Cluster Summary Information)$ 定义为: $CSI = \{n, Summary\}$, 其中 n 为类 C 的大小, $Summary$ 由分类属性中不同取值的统计频度和数值属性的质心两部分构成, 即:

$$Summary = \{ \langle Stat_i, Cen \rangle \mid Stat_i = \{(a_j, Sup_{C \mid D_i}(a_i)) \mid a_j \in D_i\}, 1 \leq i, j \leq m_C, Cen = (p_{m_C+1}, p_{m_C+2}, \dots, p_{m_C+m_N}) \}$$

1) 对象 p, q 在属性 i 上的距离 $dif(p_i, q_i)$ 定义为:

$$\text{对于分类属性或二值属性, } dif(p_i, q_i) = \begin{cases} 1 & p_i \neq q_i \\ 0 & p_i = q_i \end{cases};$$

对于连续数值属性或顺序属性, $dif(p_i, q_i) = |p_i - q_i|$;

2) 对象 p, q 间的距离 $d(p, q)$ 定义为每个属性上的距离

的平均值, 即: $d(p, q) = \sum_{i=1}^m dif(p_i, q_i) / m$ 。

3) 对象 p 与类 C 间的距离 $d(p, C)$ 定义为 p 与类 C 的摘要之间的距离: $d(p, C) = \sum_{i=1}^m dif(p_i, C \mid D_i) / m$ 。这里 $dif(p_i, C \mid D_i)$ 为 p 与 C 在属性 D_i 上的距离, 对于分类属性 D_i 其值定义为 p 与 C 中每个对象的在属性 D_i 上的距离的平均值, 即 $dif(p_i, C \mid D_i) = 1 - Sup_{C \mid D_i}(p_i) / |C|$; 对于数值属性 D_i 其值定义为 $dif(p_i, C \mid D_i) = |p_i - c_i|$ 。

4) 类 C_1 与 C_2 间的距离 $d(C_1, C_2)$ 定义为两个摘要之间的距离: $d(C_1, C_2) = \sum_{i=1}^m dif(C_1 \mid D_i, C_2 \mid D_i) / m$ 。这里 $dif(C_1 \mid D_i, C_2 \mid D_i)$ 为 C_1 与 C_2 在属性 D_i 上的距离, 对于分类属性 D_i 其值定义为:

$$dif(C_1 \mid D_i, C_2 \mid D_i) = 1 - \frac{1}{|C_1| \cdot |C_2|} \sum_{p \in C_1} Sup_{C_1 \mid D_i}(p_i) \cdot Sup_{C_2 \mid D_i}(p_i) = 1 - \frac{1}{|C_1| \cdot |C_2|} \sum_{q \in C_2} Sup_{C_1 \mid D_i}(q_i) \cdot Sup_{C_2 \mid D_i}(q_i)$$

对于数值属性 D_i 其值定义为:

$$dif(C_1 \mid D_i, C_2 \mid D_i) = |C_i^{(1)}| - |C_i^{(2)}|。$$

定义4 类 C_1 与 C_2 间的引力定义为 $g(C_1, C_2) = \frac{\ln(\ln(C_1 \cdot n + 1)) \cdot \ln(\ln(C_2 \cdot n + 1))}{d(C_1, C_2)^2}$, 这里将 $\ln(\ln(C \cdot n + 1))$ 看成类 C 的质量。特别地, 类 C 与对象 p 间引力为 $g(p, C) = \frac{\ln(\ln(C \cdot n + 1)) \cdot \ln(\ln(2))}{d(p, C)^2}$ 。

显而易见, 类间引力越大, 说明它们越相似。

为了减少数值属性不同度量单位对结果的影响, 需要对数值属性进行规范化。

2 基于引力的聚类方法 GCA

2.1 聚类算法

聚类算法 GCA 将聚类过程看成对象被已有类吸引的过程, 对象依次被吸入到对其引力最大的类中或产生一个新的类。这一方法由基本算法和增量更新模块构成。

基本算法详细描述如下:

1) 初始时, 聚类集合为空, 读入一个新的对象;

2) 以这个对象构造一个新的类;

3) 若已到数据库末尾, 则转 6), 否则读入新对象, 计算每个已有类对它的引力, 并选择最大的引力;

4) 若最大引力小于给定的阈值 r , 转 2);

5) 将该对象并入具有最大引力的类中, 并更新该类的各分类属性值的统计频度及数值属性的质心, 转 3);

6) 保存每个类的摘要信息及引力阈值 r , 结束。

GCA 方法是可以增量更新的。增量更新过程只需将基本聚类算法稍做修改即可, 即用“读入保存的已有类的摘要信息及引力阈值 r ”取代基本算法中的步骤 1)、2)。

2.2 时间和空间复杂度

聚类算法 GCA 的时间和空间复杂度依赖于数据集的大小 N , 属性个数 m , 产生 CSI 的个数及每个 CSI 的大小。为简化分析, 假定最终产生的 CSI 个数为 k , 每个分类属性 D_i 有 n_i 个不同的取值, 则最坏情况下聚类算法时间复杂度为 $O(N \cdot k(\sum_{i=1}^{m_C} n_i + m_N))$, 空间复杂度为 $O(N \cdot m + k(\sum_{i=1}^{m_C} n_i + m_N))$ 。

在聚类算法执行过程中, 生成的聚类个数是从 1 逐步扩大到 k 的, 聚类中属性值的个数也是逐步增加的。正如文献[2]指出, 分类属性通常具有小的值域, 典型的分类属性值域少于 100 个不同取值, $\sum_{i=1}^{m_C} n_i$ 会在很有限的范围内(在后面的两个

实际例子中, 对于 mushroom, $\sum_{i=1}^{m_C} n_i = 119$, 对于 CUP99, $\sum_{i=1}^{m_C} n_i = 85$)。因此, 在实际问题中, 聚类算法期望的时间复杂度为 $O(N \cdot k \cdot m)$, 空间复杂度为 $O(m \cdot (N + k))$ 。

由此可见, 时间复杂度和空间复杂度与数据集大小成线性关系, 与属性个数以及最终的聚类个数成近似线性关系, 这使得算法具有好的扩展性。

2.3 阈值 r 的选择

聚类算法中参数 r 将影响聚类的结果和算法的时间效率。 r 越大得到类的个数越多, 算法时间开销越大。当 r 小到一定值时只能得到极少的类甚至一个类; 当 r 大到一定值时, 每个类只有一个元素。 r 太大或太小都不能得到有意义、有用的聚类结果。从聚类过程直观看, r 应小于所有对象间的平均引力。为适应大数据集的情况, 我们提出一种采用抽样技术来计算阈值的方法, 具体描述如下:

1) 在数据集 D 中随机选择 N_0 对对象;

2) 计算每对对象间的引力;

3) 计算 2) 中引力的平均值 EX ;

4) 取 r 介于 $EX/2$ 与 $EX/3$ 之间。

2.4 噪音问题

数据中可能混有噪音, 当阈值 r 在合适的范围内时, 这些噪音会聚集在很稀疏的聚类中(类大小很小, 如不超过 5), 对这些稀疏聚类可以采用两种方法进行处理。其一是将这些稀疏聚类清除; 其二是将这些稀疏聚类归并到对其引力最大的聚类中。噪音处理后可提高聚类的质量。

3 实验结果

为了检验聚类算法 GCA 的有效性, 我们在文献[8]提供

的多个真实数据集上进行了测试,结果表明 GCA 是有效的,可改进最小距离原则聚类算法的聚类质量,而获得高质量的聚类结果。

3.1 mushroom 数据集

表 1 GCA 算法聚类结果

Cluster	Edible	Poisonous	Cluster	Edible	Poisonous
1	1728	0	12	192	0
2	512	0	13	96	0
3	0	256	14	48	0
4	0	192	15	48	0
5	96	0	16	0	1296
6	0	72	17	0	32
7	0	1728	18	0	8
8	192	0	19	192	0
9	288	0	20	48	0
10	768	0	21	0	36
11	0	288	22	0	8

表 2 Rock 算法聚类结果

Cluster	Edible	Poisonous	Cluster	Edible	Poisonous
1	96	0	12	48	0
2	0	256	13	0	288
3	704	0	14	192	0
4	96	0	15	32	72
5	768	0	16	0	1728
6	0	192	17	288	0
7	1728	0	18	0	8
8	0	32	19	192	0
9	0	1296	20	16	0
10	0	8	21	0	36
11	48	0			

表 3 Squeezer 算法聚类结果

Cluster	Edible	Poisonous	Cluster	Edible	Poisonous
1	0	256	13	48	0
2	512	0	14	1	72
3	768	0	15	48	0
4	96	0	16	0	32
5	96	0	17	0	8
6	192	0	18	0	859
7	1728	0	19	192	0
8	0	1296	20	288	0
9	0	192	21	0	36
10	0	288	22	31	0
11	192	0	23	0	8
12	0	869	24	16	0

mushroom 数据集有 8124 条记录,每条记录由 22 个分类属性来刻画。每条记录用来描述一种蘑菇的特性,并被标识为是有毒 P(Poisonous)还是可食用 E(Edible)。经计算得到 $EX = 35.6$,表 1 给出了 GCA 算法在 $r = 11$ 时的结果以及 ROCK 算法^[1]和 Squeezer 算法^[3]的结果,结果表明我们的算法较 ROCK 算法和 Squeezer 算法(即基于最小距离原则的聚类算法)可以产生更高质量的聚类结果。ROCK 算法和 Squeezer 算法产生的聚类,都有一个类混合有两种记录,而

GCA 算法产生的类都是由一种记录构成。实验表明,算法 GCA 的实际运行时间少于 Squeezer 算法的实际运行时间;当 $10 \leq r \leq 25$ 时,GCA 算法完全将两种类别的蘑菇聚集在了不同的类中, r 越大,得到的聚类个数越多(实际得到的聚类个数在 22 与 32 之间),当 r 增加时,大的聚类会被分割成小的聚类。

3.2 KDDCUP99

KDDCUP99 包含了约 4 900 000 条模拟攻击记录,总共有 41 个特征(属性),7 个分类特征,34 个数值型特征。我们从中随机选取一个子集 T ,包含 98 800 记录,其中 *normal* 占 19 594, *attack* 占 79 206。求得 $EX = 286.4$,表 2 给出了 $r = 95$ 时 GCA 方法与基于最小距离原则的聚类方法的在 T 上的聚类结果,其中对包含元素个数不超过 2 的聚类进行了清除。

表 4、表 5 是 GCA 与最小距离聚类方法在 CUP99 上的聚类结果。

表 4 GCA 算法的聚类结果

Cluster	normal	attack	Cluster	normal	attack
1	7	56 446	12	171	0
2	0	17 157	13	0	0
3	11 749	346	14	82	0
4	0	4 196	15	0	0
5	2 674	9	16	68	1 296
6	1 757	2	17	0	44
7	1 127	20	18	0	26
8	1 064	5	19	0	18
9	802	0	20	1	16
10	14	0	21	1	13
11	70	288	22	7	2

表 5 基于最小距离原则的聚类结果

Cluster	normal	attack	Cluster	normal	attack
1	13	56 406	16	0	163
2	0	17 099	17	41	29
3	11 448	436	18	0	44
4	4	3 931	19	0	18
5	2 496	17	20	0	17
6	1 831	16	21	2	13
7	1 129	24	22	0	14
8	1 055	8	23	10	2
9	691	55	24	2	9
10	123	292	25	1	7
11	213	136	26	4	3
12	274	9	27	1	6
13	243	35	28	6	0
14	0	237	29	6	0
15	1	180			

结果表明,GCA 比采用最小距离原理得到的聚类质量要好。一方面聚类个数要少(意味着时间效率高),另一方面正常记录与异常记录混杂的比例要少。从实际运行的时间看,GCA 的实际执行时间要少一些,尽管引力原理多了计算引力的时间,但得到的聚类个数少一些,最终导致总的执行时间也少一些。

与文献[1]、[3]、[5]比较,聚类算法 GCA 对数据集的属
(下转第 300 页)

性的分析,信息泄露率越低,该结构的应用系统就越安全。C/S 结构中就信息泄露率而言,C/S 三层结构 < C/S + 存储过程 < C/S 结构,且 C/S 与 B/S 结构的信息泄露率成倍数递减关系($C/S > 20\%$, $B/S < 5\%$),从这个角度而言,B/S 结构的安全性优于 C/S 的三层结构;但考虑到 B/S 结构的特点,客户端采用与应用无关的超文本信息查询工具——浏览器,使得任何人访问应用服务器更加简单,从而使得应用服务器受攻击的威胁增大,C/S 三层结构需要有对应得客户端才能连上应用服务器,在这方面进行比较,C/S 三层结构比 B/S 有更好的安全性。

综上所述可见:C/S 和 C/S + 存储过程的结构适用于对安全性不作要求或是对安全要求比较低的应用环境中;B/S 结构适用于安全级别要求中,网络安全性好的应用环境,在互联网环境下使用时,必须要对机密信息进行加密并且应用服务器的安全防护级别要求也需提升;C/S 三层结构适合安全性要求高的应用环境,再辅以加密措施,是一些机密数据库应用系统的最佳选择结构。

从实验中,我们还可以看出,数据库表名和数据字段名被窃听到的概率很高,而这个往往在加密中被忽视,或是考虑到响应的时延要求,没有对这些信息进行加密。解决的思路除了进行加密外,还可以制定一套内部的开发命名代号机制,不采用可理解的命名方式,而是采用代码或代号的进行命名。比如数据库中要建立访问用户表,通常方式是建立数据表:LoginUser { UserID, UserName, Password... },当窃听器抓取

到这类包的时候,也就知道该表保存了系统的用户信息;但如果采用这样的方式进行建表:Tb_010 { Fd_001, Fd_002, Fd_003 ... },即使被窃听者抓到该数据表的数据,也很难猜测到这是系统的登录用户表。通过这种方式可以降低泄露信息的风险,减少了数据传输的加解密过程,提高了系统的性能。

6 结语

本文对数据应用系统的安全性进行研究分析,提出了一种利用 Sniffer 技术量化分析数据传输安全性的方法,通过建立实验环境进行测试来验证方法的可行性,并利用实验的测试结果对目前常用的数据应用系统的传输安全性进行定量评估,客观地分析了系统的安全性。

参考文献:

- [1] LOTHIAN P, WENHAM P. Database Security in a Web Environment[J]. Information Security Technical Report, 2001, 6(2): 12 - 20.
- [2] 鲜波,张继棠,陈新安. 数据库应用系统的安全性探讨[J]. 重庆邮电学院学报, 2000, 12(1): 47 - 50.
- [3] WISEMAN S. Database Security: Retrospective and Way Forward [J]. Information Security Technical Report, 2001, 6(2): 30 - 43.
- [4] 罗朝晖,边小凡,刘铁英,等. 三层模式下信息系统的安全[J]. 计算机系统应用, 2000, (8): 17 - 20.
- [5] 张健,李焕洲. 网络嗅探原理及其检测和预防[J]. 四川师范大学学报(自然科学版), 2003, 26(1): 90 - 92.

(上接第 288 页)

性没有任何限制,既可用于纯分类属性或纯数值属性的数据集,也可用于具有混合属性的数据集,且具有更好的时间效率和聚类质量。

4 讨论

由于用到的只是引力大小的相对比较,而不是引力的绝对大小,因此省去了对结果没有影响的万有引力定律中的引力常数。前面所述引力可以看成一种特殊相似度,现有文献中的相似度仅是距离的函数,从本文可以看出,将相似度推广成距离以及类大小等因素的函数,可以更准确地度量相似性。在研究过程中我们从几个不同的角度进行了探索。

1) 最初以 C_n 作为类 C 的质量,但发现当有部分区域非常密集时,会出现类似黑洞的现象,大部分对象会被吸入,改以 $\ln(\ln(C_n + 1))$ 作为类 C 的质量则达到了比较好的效果。

2) 定义 4 中引力可推广到更一般的情况: $g(C_1, C_2) = \frac{\ln(\ln(C_1 \cdot n + 1)) \cdot \ln(\ln(C_2 \cdot n + 1))}{d(C_1, C_2)^2} (z > 0)$ 。

实验结果表明, $1 \leq z \leq 4$ 而 r 在合适的范围内时,检测结果基本稳定。

3) 将 $dissim(C_1, C_2)$ 作为 C_1 与 C_2 间差异程度的度量:

$$dissim(C_1, C_2) = \frac{d(C_1, C_2)}{\sqrt{\ln(\ln(C_1 \cdot n + 1)) \cdot \ln(\ln(C_2 \cdot n + 1))}}$$

用 $dissim(C_1, C_2)$ 与用 $g(C_1, C_2)$ 度量类 C_1 与 C_2 间的相似性应该是等价的,但当有 $d(C_1, C_2)$ 接近于 0 时,采用 $g(C_1, C_2)$ 计算的阈值不稳定,实验结果表明,采用 $dissim(C_1, C_2)$

计算阈值更稳健。

参考文献:

- [1] GUHA S, RASTOGI R, SHIM K. ROCK: A robust clustering algorithm for categorical attributes[A]. In proceedings of the 15th ICDE [C], 1999. 512 - 521.
- [2] GANTI V, GEHRKE J, RAMAKRISHNAN R. Cactus —— clustering categorical data using summaries[A]. In Proc 1999 Int Conf Knowledge Discovery and Data Mining [C], 1999. 73 - 83.
- [3] HE ZY, XU XF, DENG SC. Squeezer: an efficient algorithm for clustering categorical data[J]. Journal of Computer Science and Technology, 2002, 17(5): 611 - 624.
- [4] GUHA S, MEYERSON A, MISHRA N, et al. Clustering data streams: Theory and practice[J]. Knowledge and Data Engineering, IEEE Transactions on, 2003, 15(3): 515 - 528.
- [5] PORTNOY L, ESKIN L, STOLFO S. Intrusion Detection with Unlabeled Data using Clustering[A]. In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001) [C], Philadelphia, PA, 2001.
- [6] ESKIN E, ARNOLD A, PRERAU M, et al. A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data[Z]. In Data Mining for Security Applications, 2002.
- [7] SHENG YJ, YU MX. An Efficient Clustering Algorithm [A]. In Proc of 2004 International Conference on Machine Learning and Cybernetics [C], 2004. 8.
- [8] MERZ CJ, MERPHY P. UCI repository of machine learning databases[EB/OL]. <http://www.ics.uci.edu/mllearn/MLRRepository.html>, 2000.