文章编号:1001 - 9081(2007)02 - 0418 - 03

关联规则挖掘中对 Apriori 算法的一种改进研究

刘以安, 羊 斌
(江南大学 信息工程学院,江苏 无锡 214122)
(Lya_wx@yahoo.com.cn)

摘 要:针对 Apriori 算法寻找频繁项集问题,通过对事务数据库的布尔化表示,提出了一种直接利用布尔矩阵的行向量去搜寻频繁项集的思想。即通过向量的内积运算和判别准则逐步浓缩布尔矩阵的行向量,从而快速、直观地归纳出事务数据库的频繁项集。研究和分析表明,该方法不仅算法简单、只需扫描一次数据库,而且还具有搜索速度快、节省内存空间和处理项目集维数大等优点。对于处理超大型事务数据库和分布式事务数据库,同样也有较好的应用。

关键词:数据挖掘;关联规则;频繁项集
中图分类号: TP311 **文献标识码**:A

Research of an improved Apriori algorithm in mining association rules

LIU Yi-an, YANG Bin
(*School of Information Engineering, Southern Yangtze University, Wuxi Jiangsu* 214122, *China*)

Abstract: An enhanced Apriori algorithm which directly used the row vectors of boolean matrix for transaction databases to find out the frequent item sets was presented in this paper. It used the inner product and discriminant rule to concentrate the row vectors of boolean matrix step by step, so the frequent item sets of transaction databases can be inducted quickly and intuitively. Studies and analysis of the proposed algorithm show that it can not only scan the database once, but also has the virtues in high speed, less memory cost and handling with large item set dimensions. It can also be well applied to super transaction database and distributed transaction database.

Key words: data mining; association rules; frequent item set

0 引言

关联规则挖掘是数据挖掘中的一个重要研究内容。但在众多的关联规则挖掘中,Apriori 算法[1]是最基本也是最著名的一种。算法的核心思想是基于频集理论的一种递推方法,目的是从数据库中挖掘出那些支持度和信任度都不低于给定的最小支持度阈值和最小信任度阈值的关联规则。Apriori 算法通常分为两步:1)基于支持度,产生频繁项集;2)基于可信度,产生强关联规则。其核心是由 1)生成的频繁项集。而 Apriori 算法的不足是需要对数据库进行多次扫描,候选集项目数多,内存利用率低,以致影响运行效率。目前,众多学者针对 Apriori 算法的不足,提出了许多较好的改进或扩展方法,如 DHP 方法[2]、Partition 法[3]、频繁闭项集法[4]、FP-Growth 算法[5]、闭包项集格[6]、TBAR 算法[7]、动态剪枝[8]等。尽管这些算法各具优点且挖掘的性能和效率均明显高于传统的 Apriori 算法,但总的来说,算法仍较复杂。特别地,当项目集维数较大时,这些算法的挖掘效能仍然较低。

为了方便、快速地从事务数据库中挖掘出频繁项集,本文从 Apriori 算法寻找频繁项集的过程出发,提出了一种简单、实用的有效挖掘方法。首先,将事务数据库映射到布尔矩阵,然后针对布尔矩阵的行向量应用向量内积运算,找出频繁项集可能存在的行以达到逐步浓缩布尔矩阵行向量的目的,最后从浓缩的布尔矩阵中快速、直观地归纳出事务数据库要找的频繁项集。

1 Apriori 算法简介

Apriori 算法是一种寻找频繁项集的基本算法,即找出所有支持度不小于给定 min sup 的项集。其基本原理是使用一种称作逐层搜索的迭代方法,即用 k - 项集去探索($k+1$) - 项集。

设 $I=\{I_1,I_2,\cdots,I_m\}$ 为事务数据库 D 中 m 个不同项目组成的集合,其中的每一项目 $I_i(i=1,2,\cdots,m)$ 相当于一种商品。$W=\{T_1,T_2,\cdots,T_n\}$ 是一组事务集,W 中的每个事务 $T_i(i=1,2,\cdots,n)$ 是一组商品,$T_i \subseteq T$。每个事务 T 都有唯一标识 TID。项目集中项目的个数称为项目集的维数或长度,若项目集的长度为 k,称为 k - 项集。则对于任一给定的事务数据库 D,其频繁项集产生的过程可描述[9]为:

1)先计算所有的 1 - 项集,记为 C_1。找出大于或等于给定最小支持度 min sup 的所有常用的 1 - 项集,记为 L_1;

2)根据常用 1 - 项集确定候选 2 - 项集的集合,记为 C_2。从 C_2 找出大于或等于给定最小支持度 min sup 的所有常用 2 - 项集,记为 L_2;

3)由常用 2 - 项集确定候选 3 - 项集的集合,记为 C_3。从 C_3 找出大于或等于最小支持度 min sup 的所有常用 3 - 项集,记为 L_3。如此下去,直到不能找到更高维的频繁项集为止。

显然,Apriori 算法需要对数据库进行多次扫描,且产生的

收稿日期:2006 - 08 - 31

作者简介:刘以安(1963 -),男,江苏涟水人,副教授,博士,主要研究方向::数据挖掘、数据融合、雷达对抗; 羊斌(1983 -),男,浙江磐安人,硕士研究生,主要研究方向:数据挖掘、数据融合.

中间候选项集数较多。特别地,当项目个数较多时,该算法的复杂度将呈指数级增长,影响运行效率。

2　事务数据库的布尔矩阵表示

对于任一给定的事务数据库 D,令:

$$f: D \to R \tag{1}$$

其中:

$$R = f(D) = (r_{ij})_{n\times m}$$

这里:

$$r_{ij} = \begin{cases} 1, & I_j \in T_i \\ 0, & I_j \notin T_i \end{cases} \quad i = 1,2,\cdots,n; j = 1,2,\cdots,m$$

于是,事务数据库 D 经一次扫描后,就可在 f 的作用下映射成布尔矩阵 $\boldsymbol{R}$。例如,对于文献[10]所给的一个事务数据库 D,如图 1 所示,可映射成图 2 所示的布尔矩阵 $\boldsymbol{R}$。

TID	List of item-1ds
100	I_1,I_2,I_5
200	I_2,I_4
300	I_2,I_3
400	I_1,I_2,I_4
500	I_1,I_3
600	I_2,I_3
700	I_1,I_3
800	I_1,I_2,I_3,I_5
900	I_1,I_2,I_3

图 1　事务数据库

$$\Rightarrow \quad \boldsymbol{R}=\begin{bmatrix} 1&1&0&0&1\\ 0&1&0&1&0\\ 0&1&1&0&0\\ 1&1&0&1&0\\ 1&0&1&0&0\\ 0&1&1&0&0\\ 1&0&1&0&0\\ 1&1&1&0&1\\ 1&1&1&0&0 \end{bmatrix}$$

图 2　事务数据库的布尔矩阵表示

3　向量内积与算法改进

由上分析知,对于任意一个事务数据库只需扫描一次,即可获得与该数据库相对应的一个布尔矩阵。故对事务数据库的挖掘问题就可转化为对其布尔矩阵的分析。

3.1　向量内积

由高等代数关于向量空间内积的定义[11]知,对于任意两个 m 维向量 $\boldsymbol{\alpha} = (x_1,x_2,\cdots,x_m)$ 和 $\boldsymbol{\beta} = (y_1,y_2,\cdots,y_m)$,则 $\boldsymbol{\alpha}$ 和 $\boldsymbol{\beta}$ 的内积定义为:

$$< \boldsymbol{\alpha},\boldsymbol{\beta} > = \sum_{i=1}^{m} x_i y_j \tag{2}$$

3.2　算法改进

由 Apriori 算法搜寻频繁项集的结果不难发现,每个频繁项集中包含的项目完全出现在事务数据库 D 中的某些元组(记录)中。假设事务数据库 D 对应的布尔矩阵为 $\boldsymbol{R}$,记 $\boldsymbol{R} = (\alpha_1,\alpha_2,\cdots,\alpha_n)^{\mathrm{T}}$,其中 T 为转置,$\alpha_i$ 为 $\boldsymbol{R}$ 在实数域上的 m 维行向量,$i = 1,2,\cdots,n$。记 $I = \{I_1,I_2,\cdots,I_m\}$ 为 D 中 m 个不同项目组成的集合,如果已知 I 的某个子集 $T_p \subseteq I$ 为 D 的 k - 频繁项集,令 α 为 T_P 按式(1)所映射的 m 维行向量,则有:

$$< \alpha,\alpha_i > \leqslant < \alpha,\alpha > = k,\ i = 1,2,\cdots,n \tag{3}$$

这表明,事务数据库 D 中的频繁项集所对应的 m 维行向量与布尔矩阵 $\boldsymbol{R}$ 中的每个行向量作内积,其内积和均不会超过频繁项集所含的项目个数。故在用布尔矩阵 $\boldsymbol{R}$ 的行向量去搜寻事务数据库 D 的可能频繁项集时,应通过向量内积和,先设法找出频繁项集对应 $\boldsymbol{R}$ 中可能存在的行向量,然后将那些含非零个数比它少的行向量以及与它相同的行向量加以标记,这样在后续频繁项集搜寻中就可跳过这些标记过的行,以提高搜寻的速度。对于频繁项集在 $\boldsymbol{R}$ 中可能对应的行向量的确定,可由频繁项集的定义按如下规则判断:

1)针对布尔矩阵 $\boldsymbol{R}$ 的第 i 行向量 α_i($i = 1,2,\cdots,n$ 且 $i \neq$ 标记符),统计 $< \alpha_i,\alpha_j > = < \alpha_i,\alpha_j >$ 的个数 β_i。这里,$j = 1,2,\cdots,n$,$j \neq i$ 且 $j \neq$ 标记符;

2)判断 $\beta_i + 1 \geqslant \min\ \sup$ 是否成立。若成立,则 α_i 为频繁项集可能对应的行向量;否则,标记该 i 行。这里加 1,表示 α_i 自身在内。

又由于 Apriori 算法是利用频繁项集向下封闭性的一种宽度优先算法,即频繁项集的子集必是频繁项集。故对布尔矩阵 $\boldsymbol{R}$,当按上述方法对其所有行向量判断完一遍后,删除 $\boldsymbol{R}$ 中所有带标记的行。然后,再从剩余的行向量中删除那些行向量所对应的非零元素完全包含在另一行向量中的行,最后剩下的行向量就是我们要找的频繁项集所对应的行向量。具体的算法步骤描述如下:

1)扫描事务数据库 D,建立其对应的布尔矩阵 $\boldsymbol{R} = (\alpha_1,\alpha_2,\cdots,\alpha_n)^{\mathrm{T}}$,并给出用户期望的最小支持度 min sup,置行循环变量 $i = 1$;

2)先对布尔矩阵 $\boldsymbol{R}$ 的各个列向量,求自内积。删除 $\boldsymbol{R}$ 中自内积的和小于最小支持度 min sup 的列(项),所形成的新矩阵仍记为 $\boldsymbol{R}$;

3)按上述判断规则的描述计算 β_i,并判断 $\beta_i + 1 \geqslant \min\ \sup$ 是否成立。

4)若成立,则 α_i 即为频繁项集对应的可能行向量,同时标记出 $\boldsymbol{R}$ 中其他行所含非零个数比 $< \alpha_i,\alpha_i >$ 小的行,然后继续搜索下面没有标记的行,并对 i 作相应的修改;若不成立,则标记此行,然后继续下面没有标记的行进行搜索,同样对 i 作相应的修改;

5)若 $i \leqslant n$,则转 3)。否则转 6);

6)删除 $\boldsymbol{R}$ 中所有带标记的行。最后,将剩下的行向量按它们所含非零元素顺序给出相应的频繁项集。

显然,从上述描述中可知该算法具有如下的优点:

1)由于只使用了简单的向量内积运算和判断规则,故算法既简单,又能较好地处理项目集维数较大的情况;

2)只需扫描一次数据库,且频繁项集的搜索除了要增加标记位外,整个过程均在布尔矩阵 $\boldsymbol{R}$ 所开辟的内存空间中完成,不必考虑内存空间分配和候选项过多等问题;

3)在该算法的运行过程中,由于不断地将一些行加以标记,从而逐步浓缩了对布尔矩阵行向量的搜索,有效提高了搜索速度。

3.3　示例

为了直观地说明上述算法过程,这里以图 1 所示的事务数据库为例。先将事务数据库 D 按项目 I_1,I_2,I_3,I_4,I_5 顺序映射成布尔矩阵 $\boldsymbol{R}$,如图 2 所示。设给定的最小支持度 min sup = 2,记 $\boldsymbol{R} = (\alpha_1,\alpha_2,\cdots,\alpha_9)^{\mathrm{T}}$,则整个搜索过程如图 3 所示。

$\boldsymbol{R}$	I_1	I_2	I_3	I_4	I_5	β_i+1	标记
α_1	1	1	0	0	1	2	
α_2	0	1	0	1	0		*
α_3	0	1	1	0	0		*
α_4	1	1	0	1	0	1	*
α_5	1	0	1	0	0		*
α_6	0	1	1	0	0		*
α_7	1	0	1	0	0		*
α_8	1	1	1	0	1	1	*
α_9	1	1	1	0	0	2	
1-项集	6	7	6	2	2		

图 3　频繁项集的搜寻示意图

从图 3 可以直观地看出,算法的第一步是将 $\boldsymbol{R}$ 的各列自内积,以确定 1 - 项集;第二步是计算 α_1 向量对应的 $\beta_1 + 1$,通过判断说明 α_1 为频繁项集对应的可能行向量,同时标记比 α_1 中非零个数小的向量 $\alpha_2,\alpha_3,\alpha_5,\alpha_6$ 和 α_7 所对应的行;第三步算法从第四行 α_4 开始,计算 $\beta_4 + 1$,由于 $\beta_4 + 1 <$ min sup,则

标记 α_4 对应的行;第四步算法从第八行 α_8 开始,计算 β_8+1,同样因 $\beta_8+1<\min\sup$ 而将 α_8 对应的行标记;第五步算法从第九行 α_9 开始,计算 β_9+1,因 $\beta_9+1\geqslant\min\sup$,则 α_9 为频繁项集对应的可能行向量。最后删除所有的标记行,只剩下 α_1 和 α_9 对应的行,且这两个行向量没有非零元素间的包含关系,故从中可直接获得事务数据库 D 对应 2 个频繁项集,分别为 $\{I_1I_2I_3\}$ 和 $\{I_1I_2I_5\}$。这与 Apriori 算法产生的结果完全一致。

4 超大型事务和分布式事务数据库挖掘

随着计算机的普及和网络、数据库技术的快速发展,目前数据库正朝着超大型、多维化和分布式方向发展。传统的挖掘算法已很难适应大规模、可扩展挖掘的需要。为此,进一步探索适应数据库发展的数据挖掘方法和工具仍是十分必要的。这里,仍应用本文提出的改进算法对超大型事务数据库和分布式事务数据库进行研究和分析,以进一步说明该方法的适应性和有效性。

4.1 超大型事务数据库的挖掘

由于超大型数据库存在内存瓶颈问题,所以在数据准备阶段,应按照用户要求或计算机内存的容量情况,先对数据库进行分割,然后进行分段扫描。假设事务数据库 D 被分割成 N 份,记为 $D_1,D_2,\cdots,D_N$。则由式(1),有 N 个布尔矩阵 $\boldsymbol{R}_1,\boldsymbol{R}_2,\cdots,\boldsymbol{R}_N$ 分别与 $D_1,D_2,\cdots,D_N$ 对应。于是,对超大型事务数据库 D,其频繁项集的生成过程可描述为:

1) 设定超大型事务数据库 D 的分割份数 N 和各份的大小。置循环变量 $i=1$,给出用户期望的最小支持度 min sup;

2) 读入事务数据库 D 的第 i 份 D_i 的同时,将其映射成布尔矩阵 $\boldsymbol{R}_i$;

3) 由上述的算法步骤,对 $\boldsymbol{R}_i$ 按 $\min\sup_i=\min\sup\times\frac{|D_i|}{|D|}$ 作为 D_i 的局部最小支持度,找出 D_i 中的频繁项集在 $\boldsymbol{R}_i$ 中所对应的行向量,并将这些行向量重新保存,释放原 $\boldsymbol{R}_i$ 开辟的内存空间。对新保存的矩阵仍记为 $\boldsymbol{R}_i$。这里,$|D_i|$ 表示为 D_i 中所含的元组数;$|D|$ 为超大型事务数据库的元组数;

4) 令 $i=i+1$。如果 $i\leqslant N$,则转至 2) 重复执行。否则转至 5);

5) 将 D_i 中各频繁项集所对应的布尔矩阵 $\boldsymbol{R}_i,i=1,2,\cdots,N$,重新组合形成一新的 $\boldsymbol{R}$ 布尔矩阵,即 $\boldsymbol{R}=(R_1,R_2,\cdots,R_N)^{\mathrm{T}}$。然后,再应用 3.2 节描述的算法步骤,对 $\boldsymbol{R}$ 按 min sup 最小支持度,找出超大型事务数据库 D 的频繁项集所对应的行向量,进而得到 D 最终形成的频繁项集。

4.2 分布式事务数据库的挖掘

对于来自不同地点数据源所形成的分布式数据库,其局部频繁项集与全局频繁项集之间存在一些有价值的关联[9]:

1) 一个局部的频繁项集对于全局来说,不一定是频繁项集;

2) 每一个全局频繁项集至少在某一地点是局部频繁项集。

故根据这些关联,可将分布式事务数据库 D 的挖掘问题研究转化为类似超大型事务数据库的挖掘问题。

假设事务数据库 D 存放在分布式系统的 N 个地点,各地点上的数据分别为 $D_1,D_2,\cdots,D_N$ 且对应的布尔矩阵是 $\boldsymbol{R}_1,\boldsymbol{R}_2,\cdots,\boldsymbol{R}_N$。若令 min sup 为 D 的全局支持度,则每一 D_i 的局部支持度为:

$$\min\sup_i=\min\sup\times\frac{|D_i|}{|D|},\ i=1,2,\cdots,N$$

在生成分布式事务数据库 D 的频繁项目集时,可将分布式数据库 D 看成一超大型事务数据库,而将分布在各地点上的数据集 $D_1,D_2,\cdots,D_N$ 视为 D 上的 N 个分割。这样,对分布式事务数据库 D 的频繁项集搜寻,可采用上节的思想先找出每个局部地点 $i(i=1,2,\cdots,N)$ 频繁项集对应的行向量所形成的布尔矩阵 $\boldsymbol{R}_i,i=1,2,\cdots,N$,然后再将每一地点频繁项集对应的布尔矩阵向其他地点广播,生成新的全局布尔矩阵 $\boldsymbol{R}$。最后,$\boldsymbol{R}$ 按全局最小支持度 min sup,找出分布式事务数据库 D 的频繁项集对应的行向量,获得 D 的全局频繁项集。

5 结语

本文提出的从事务数据库所对应的布尔矩阵出发,直接利用行向量内积的方法去搜寻频繁项集问题,不仅算法简单、只需扫描一次数据库,而且还具有方便处理超大型事务数据库和分布式事务数据库等方面的优点。这对于探索和解决当今时代所面临的数据爆炸而信息贫乏等问题,仍具有一定的参考意义。

参考文献:

[1] AGRAWAL R, IMIELINSKI T, SWAMI A. Mining association rules between sets of items in large database[A]. In Proc. of the ACM SIGMOD Intl Conf. on Management of Data[C]. Washington, D. C., 1993. 207 - 216.

[2] PARK JS, CHEN MS, YU PS. An effective hash based algorithm for mining association rules[A]. ACM SIGMOD International Conference Management of Data[C]. 1995. 175 - 186.

[3] SAVASERE A, OMIECINSKI E, NAVATHE S. An efficient algorithm for mining association rules in large database[A]. In: Proc. of 21th Intl Conf. on Very Large DataBase[C]. Zurich, Swizerland, 1995. 432 - 443.

[4] PASQUIER N, BASTIDE Y, TAOUIL R, *et al*. Discovering frequent closed item sets for association rules[A]. ICDT'99, Israel[C]. 1999. 398 - 416.

[5] HAN J, PEI J, YIN Y. Mining frequent patterns without candidate generation[A]. In: Proceedings of the 2000 ACM SIGMOD Internal Conference on Management of Data[C]. Dallas, Texas: ACM Press, 2000. 1 - 12.

[6] PASQUOER N, BASTIDE Y, TAOUIL R. Effictient mining of association rules using closed item set lattices[J]. Information System, 1999, 24(1): 25 - 46.

[7] BERZAL F, CUBERO J-C, MARIN N. TBAR: An efficient method for association rule mining in relational databases[J]. Data & Knowledge Engineering, 2001, 37: 47 - 64.

[8] 皮德常, 秦小麟, 王宁生. 基于动态剪枝的关联规则挖掘算法[J]. 小型微型计算机系统, 2004, 25(10): 1850 - 1852.

[9] 史忠植. 知识发现[M]. 北京: 清华大学出版社, 2002.

[10] HAN J-W, KAMBR M. Data mining concepts and techniques[M]. Beijing: Higher Education Press, 2001. 255 - 279.

[11] 张禾瑞, 赫炳新. 高等代数[M]. 第 3 版. 北京: 高等教育出版社, 1993.