

Apriori 算法的优化方法

陈 伟

(淮南联合大学 计算机系, 安徽 淮南 232038)

摘 要:关联规则是数据挖掘的主要技术之一,是指从一个大型的数据集中发现有趣的关联或相关关系,即从数据集中识别出频繁项集,然后再利用这些频繁项集创建描述关联规则的过程。频繁项集挖掘是关联规则挖掘的主要步骤,在频繁项集挖掘中,需要大量进行两个操作:判断两个 k -项集是否是前 $k-1$ 项相同且最后一项不同,即连接步;判断一个项集是否为另一个项集的子集,即剪枝步,通过减少连接操作和剪枝操作的循环次数,以此来提高 Apriori 算法的效率。

关键词:关联规则; Apriori 算法; 频繁项集

中图分类号: TP301.6

文献标识码: A

文章编号: 1673-629X(2009)06-0080-04

Method of Apriori Algorithm Optimization

CHEN Wei

(Dept. of Computer, Huainan Union University, Huainan 232038, China)

Abstract: Association rule is one of the key technologies of data mining to find the funny association or relationship from the given dataset, it is to say, association rule is to extract frequent item set from the dataset at first, and then establish the process of description of association rule according to frequent item. The frequent itemsets mining is the main steps of association rule mining. In the frequent itemsets mining, need to carry out a large number of two operations: judge the two k -item sets of whether the former $k-1$ of the same and the last different, that is step-by-step connection; to determine whether an item set for another subset of the set, that is step-by-step pruning, can reduce the times of cycle the connect operation and the pruning operation, in order to improve the efficiency of Apriori algorithm.

Key words: association rule; Apriori algorithm; frequent itemsets

0 引言

在 Apriori 算法中需要大量进行这样两个操作:判断两个 k -项集中是否前 $k-1$ 项相同且最后一项不同,即连接步;判断一个 k -项候选集的所有 $k-1$ 子集是否都是 k -项频繁集,即剪枝步。假定事务数据库中各记录的项目均已按字典排序。可以利用项集之间有序的特点,从减少算法中这两个操作的执行次数的角度来达到优化算法的目的。

1 关联规则的简单描述

关联规则是描述数据库中一组数据项之间的某种潜在关系的规则。关联规则形式简洁、易于解释和理解,并可以有效地捕捉数据间的重要关系,从大型数据库中挖掘关联规则问题已成为数据挖掘中最成熟、最

重要、最活跃的研究内容。

一般地,关联规则挖掘是指从一个大型的数据集 (Dataset) 中发现有趣的关联 (Association) 或相关 (Correlation) 关系,即从数据集中识别出频繁出现的属性值集 (Sets of Attribute Values), 也称为频繁项集 (Frequent Itemsets, 简称频繁集), 然后再利用这些频繁集创建描述关联规则的过程。

关联规则可形式化定义为:

设 $I = \{i_1, i_2, \dots, i_m\}$ 是由 m 个不同的项组成的集合。给定一个事务数据库 D , 其中每一个事务 T 是 I 中一组项的集合, 即 $T \subseteq I$, T 有一个唯一的标示符 TID。若项集 $X \subseteq I$ 且 $X \subseteq T$, 则事务 T 包含项集 X 。

关联规则是形如 $X \Rightarrow Y$ 的蕴涵式, 其中 $X \subseteq T$, $Y \subseteq T$, 并且 $X \cap Y = \emptyset$, 如果 D 中事务包含 $X \cup Y$ 的百分比为 S , 则称 S 为关联规则 $X \Rightarrow Y$ 的支持度, 它是概率 $P(X \cup Y)$, 如果 D 中包含 X 的事务同时也包含 Y 的百分比为 C , 则称 C 为关联规则 $X \Rightarrow Y$ 的置信度, 它是条件概率 $P(Y/X)$ 。习惯上将关联规则表示为 $X \Rightarrow Y (S\%, C\%)$ 。关联规则的发现任务

收稿日期: 2008-09-21; 修回日期: 2009-01-04

基金项目: 2007 年安徽省高校青年教师资助计划项目 (2007jq1197)

作者简介: 陈 伟 (1975-), 女, 安徽六安人, 讲师, 研究方向为数据库、数据挖掘。

或问题就是在事务数据库 D 中找出所具有用户给定的最小支持度阈值 (\min_sup) 和最小置信度阈值 (\min_cof) 的关联规则, 即这些关联规则的支持度和置信度分别不小于最小支持度和最小置信度。

如果项集的支持度满足最小支持度 \min_sup , 则它称之为频繁项集 (Frequent Itemset)^[1]。

所以关联规则的挖掘一般分为以下两个过程:

(1) 找出所有的频繁项集, 也就是找出事务数据库中所有大于等于用户指定的最小支持度的数据项集, 即具有最小支持度的数据项集。

(2) 由频繁项集产生强关联规则: 根据定义, 这些规则必须满足最小支持度和最小置信度。

在上述两个步骤中, 第一步是挖掘关联规则的关键步骤。关联规则挖掘的总体性能由该步骤的性能所决定。所以, 现有的研究都集中在第一个步骤上, 也就是对频繁项集的挖掘处理上。

2 Apriori 算法

关联规则挖掘有多种分类方法: 单维、单层和布尔关联规则挖掘。它们都是最简单形式的关联规则挖掘, 最著名、最有影响的关联规则挖掘算法是 R. Agrawal 等人提出的 Apriori 算法, 该算法是一种挖掘布尔关联规则频繁项集的算法。算法基于频繁项集性质的先验知识, 使用一种称为逐层搜索的迭代方法来找出所有的频繁项集。

Apriori 算法通过对数据库 D 的多趟扫描来发现所有的频繁项目集。在第一趟扫描数据库时, 对项集 I 中的每一个数据项计算其支持度, 确定出满足最小支持度的频繁 1 项集的集合 L_1 , 然后, L_1 用于找频繁 2 项集的集合 L_2 , 如此下去……在后续的第 k 趟扫描中, 首先以 $k-1$ 趟扫描中所发现的含 $k-1$ 个元素的频繁项集的集合 L_{k-1} 为基础, 生成所有新的候选项目集 C_k (Candidate Itemsets), 即潜在的频繁项目集, 然后扫描数据库 D , 计算这些候选项目集的支持度, 最后从候选集 C_k 中确定出满足最小支持度的频繁 k 项集的集合 L_k , 并将 L_k 作为下一趟扫描的基础。重复上述过程直到再也发现不了新的频繁项目集^[2]。

Apriori 算法的基本描述如下:

输入: 事务数据库 D ;

最小支持度计数阈值 \min_sup 。

输出: D 中的频繁项集 L 。

(1) $L_1 = \text{find_frequent_1_itemsets}(D)$;

(2) For ($k = 2; L_{k-1} \neq \emptyset; k++$) do begin

(3) $C_k = \text{apriori_gen}(L_{k-1}, \min_sup)$; /* 生成候

选 k - 项集 */

(4) For all transactions $t \in D$ do begin

(5) $C_t = \text{subset}(C_k, t)$; /* 候选集 C_k 中提取包含在事务 t 中的候选 k - 项集 */

(6) For all Candidates $C \in C_t$ do

(7) $C.\text{Count}++$;

(8) End

(9) $L_k = \{C \in C_k \mid C.\text{Count} \geq \min_sup\}$;

(10) End

(11) return $L = \bigcup_k L_k$;

函数 Apriori_gen 按如下方式分两步进行工作:

● 连接步

Procedure $\text{apriori_gen}(L_{k-1}; \min_sup)$

1) for each itemset $L_1 \in L_{k-1}$

2) for each itemset $L_2 \in L_{k-1}$

3) if $((L_1[1] = L_2[1]) \wedge (L_1[2] = L_2[2]) \wedge \dots \wedge (L_1[k-2] = L_2[k-2]) \wedge (L_1[k-1] < L_2[k-1]))$ then {

4) $C = L_1 \oplus L_2$;

5) if $\text{has_infrequent_subset}(C, L_{k-1})$ then

6) delete C ;

7) else add C to C_k ;

8) }

9) return C_k ;

● 剪枝步

Procedure $\text{has_infrequent_subset}(C, L_{k-1})$

1) for each $(k-1)$ - subset s of C

2) if $s \notin L_{k-1}$ then

3) return true;

4) return false;

C_k 中的成员可以是频繁的也可以是不频繁的, 但所有的频繁 k - 项集都包含在 C_k 中。如果确定 C_k 中每个候选的计数, 从而确定 L_k , 那么涉及的计算量就很大。为压缩搜索空间, 可以用 Apriori 性质: 任何非频繁的 $(k-1)$ - 项集都不可能是频繁 k - 项集的子集。因此, 如果一个候选 k - 项集的 $(k-1)$ - 项集不在 L_{k-1} 中, 则该候选也不可能是频繁的, 从而可以从 C_k 中删除。

下面来举例说明以上阐述的过程。设事务数据库 D , 见表 1。

假定要求最小支持度计数为 2, Apriori 算法执行过程如下:

(1) 算法第一次扫描数据库, 确定候选 1 - 项集及各项集的支持度计数 C_1 , 如图 1 所示。

表 1 事务数据库

TID	Item
1	1,3,4
2	2,3,5
3	1,2,3,5
4	2,5

(2) 利用 $L_1 \oplus L_1$ 来产生候选 2-项集 C_2 , 由 C_2 确定频繁 2-项集, 如图 2 所示。

(3) 利用 $L_2 \oplus L_2$ 进行连接操作, 获得候选 3-项集 $C_3 = \{2,3,5\}$ 。根据 Apriori 性质: 一个频繁项集的所有子集也是频繁的, $\{2,3,5\}$ 的 2 项子集是 $\{2,3\}$, $\{2,5\}$ 和 $\{3,5\}$, 所有 2 项子集都是 L_2 的元素, 因此是频繁的。结果如图 3 所示。

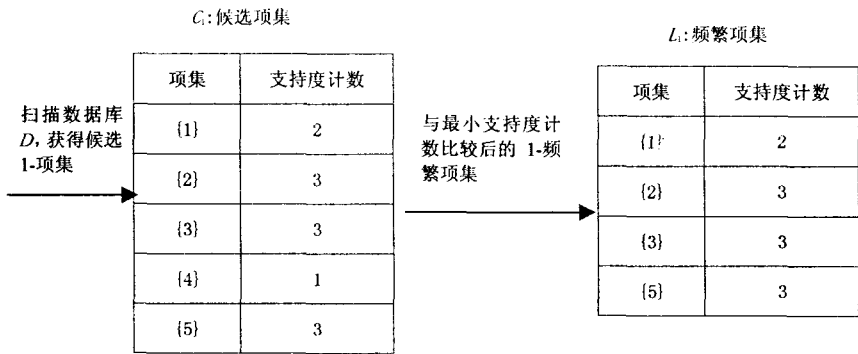


图 1 产生 1 频繁项集

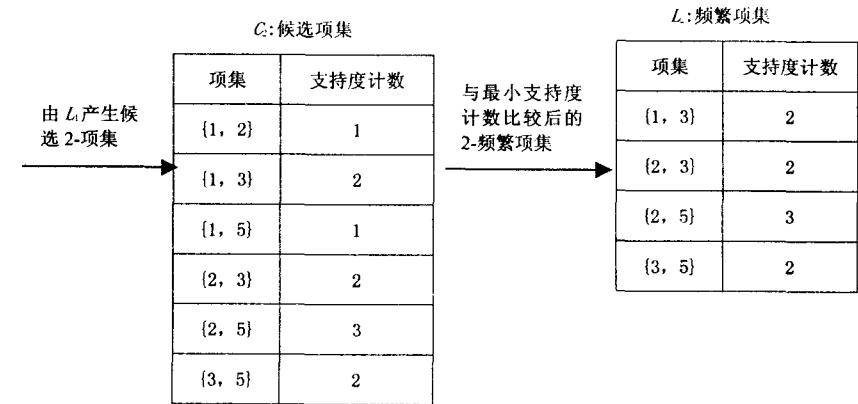


图 2 产生 2 频繁项集

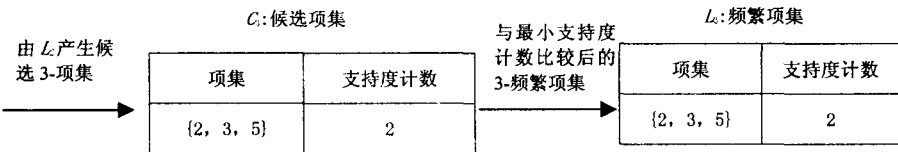


图 3 产生 3 频繁项集

(4) 利用 $L_3 \oplus L_3$ 进行连接操作得到 C_4 , 根据 Apriori 性质可知 $C_4 = \emptyset$, 算法至此结束。

在 Apriori 算法中连接步和剪枝步是频繁操作的两个步骤, 下面就从这两个步骤入手, 来提高算法的效率。

3 减少连接步骤的执行次数

由于已经设定各事务项目按字典排序, 其中的任一个 k -项集 L , 有 $L[1] < L[2] < \dots < L[k]$ 。所以对于两个 $(k-1)$ -项频繁集 L_a 和 L_b ($a < b$), 若 L_a 和 L_b 不符合连接条件, 则 L_a 与 L_b 之后的所有 $(k-1)$ -项集都不能满足连接条件。所以只要 L_a 与 L_b 不能连接, 就不需要再判断 L_a 与 L_b 之后的所有 $(k-1)$ -项集是否能连接, 从而减少循环的次数^[3,4]。改进算法:

```

Procedure apriori-gen ( $L_{k-1}$ ; min_sup)
  for each itemset  $L_a \in L_{k-1}$ 
    for each itemset  $L_b \in L_{k-1}$ 
      if  $((L_a[1] = L_b[1]) \wedge (L_a[2] = L_b[2]) \wedge \dots \wedge (L_a[k-2] = L_b[k-2])) \wedge (L_a[k-1] < L_b[k-1])$  then
         $C = L_a \oplus L_b$ ;
        if has_infrequent_subset( $C, L_{k-1}$ ) then
          delete C;
        else add C to  $C_k$ ;
      else break;
  return  $C_k$ ;
  
```

4 减少剪枝步骤的执行次数

对于一个 k -项候选集的任意一个子集 $(k-1)$ -项集 L_a 和一个 $(k-1)$ -项频繁集 L_b , 若 $L_a[i] = L_b[i]$, 则 L_a 是频繁的, 结束判断; 若 $L_a[i] > L_b[i]$, 则继续和下一个 $(k-1)$ -项频繁集比较, 如此下去; 若 $L_a[i] < L_b[i]$, 由于各事务项目按字典排序, 则 L_a 与 L_b 后的所有 $(k-1)$ -项频繁集都不会相同。所以只要 $L_a[i] < L_b[i]$, 就不需要再判断 L_a 是否与 L_b 后的 $(k-1)$ -项集相同。由此就能判定 L_a 不是频繁项集。由 Apriori 算法性质: 频繁

项集的所有非空子集都必须是频繁的, 所以该 k -项候选集也不是频繁项集, 从而删除它。该步操作中大大减少了判断候选项集的任一子集是否是频繁项集的执行次数^[5~7]。改进的算法为:

```

Procedure has_infrequent_subset( $C, L_{k-1}$ )
  
```

```

for each  $L_b \in L_{k-1}$ 
  if  $L_a \subset C$  then //  $L_a[i] \in L_a$  and  $L_b[i] \in L_b$ 
    {if ( $L_a[i] = L_b[i]$ ) then
      {  $L_a \in L_{k-1}$ ; break; }
    else if ( $L_a[i] > L_b[i]$ ) then
      continue;
    else break;
  }
if  $L_a \in L_{k-1}$  then
  return true; return false;

```

5 举例

下面举例说明^[8,9]。

(1) 连接步: 假设有 6 个 2 - 项频繁集: $L_1 = \{1, 2\}$, $L_2 = \{1, 3\}$, $L_3 = \{1, 5\}$, $L_4 = \{2, 3\}$, $L_5 = \{2, 4\}$, $L_6 = \{2, 5\}$ 。 L_1 和 L_2 、 L_1 和 L_3 满足连接条件, 可以连接。 L_1 和 L_4 不满足连接条件, 不能连接。依照改进的算法, L_1 和 L_4 之后的所有频繁项集都不满足连接条件, 从而减少了 L_1 和 L_5 、 L_1 和 L_6 的判断。

(2) 剪枝步: 假设有一个 3 - 项候选项集: $C = \{1, 3, 5\}$, 4 个 2 - 项频繁集: $L_1 = \{1, 3\}$, $L_2 = \{2, 3\}$, $L_3 = \{2, 5\}$, $L_4 = \{3, 5\}$ 。 C 的 2 - 项子集为: $C_1 = \{1, 3\}$, $C_2 = \{1, 5\}$, $C_3 = \{3, 5\}$ 。第一步在 2 - 项频繁集中寻找 C_1 , 首先 C_1 和 L_1 比较: $C_1[1] = L_1[1]$, $C_1[2] = L_1[2]$, 所以 $C_1 = L_1$ 。第二步在 2 - 项频繁集中寻找 C_2 , 首先 C_2 和 L_1 比较: $C_2[1] = L_1[1]$, $C_2[2] > L_1[2]$, 接着 C_2 和 L_2 比较: $C_2[1] < L_2[1]$, 按照改进的算法, 以后的都不用比较, 也就确定该候选项集 C 的子集 C_2 不是频繁项集, 由 Apriori 算法性质确定该候选项集 C 也不是频繁项集, 所以删除它。

(上接第 79 页)

- [28] Osher S, Sethian J. Fronts propagating with curvature-dependent speed: algorithms based on the Hamilton-Jacobi formulation[J]. Journal of Computational Physics, 1988, 79(1): 12 - 49.
- [29] 陈波, 赖剑煌, 马建华. 一种耦合的活动轮廓模型及其在图像分割中的应用[J]. 中国图象图形学报, 2007, 12(3): 444 - 449.
- [30] 谢勤彬, 罗代升. 基于改进活动轮廓模型的超声图像分割[J]. 科学技术与工程, 2007, 7(8): 1638 - 1641.
- [31] 陈波, 赖剑煌. 用于图像分割的活动轮廓模型综述[J]. 中国图象图形学报, 2007, 12(1): 11 - 20.
- [32] Bezdek J C. Pattern recognition with fuzzy objective function algorithms [M]. Norwell, MA, USA: Kluwer Academic Publishers, 1981.
- [33] 刘华军, 任明武, 杨静宇. 一种改进的基于模糊聚类的图像分割方法[J]. 中国图象图形学报, 2006, 11(9): 1312 - 1316.
- [34] 秦昆, 徐敏. 基于云模型和 FCM 聚类的遥感图像分割方法[J]. 地球信息科学, 2008, 10(3): 302 - 307.
- [35] 郑洪英. 数据挖掘聚类算法的分析和应用研究[D]. 重庆: 重庆大学, 2002.
- [36] 邵锐, 巫兆聪, 钟世明. 基于粗糙集的 K - 均值聚类算法在图像分割中的应用[J]. 测绘信息与工程, 2005, 30(5): 1 - 2.
- [37] 许海洋, 王万森. 基于 SOM 神经网络和 K - 均值算法的图像分割[J]. 计算机工程与应用, 2005(21): 38 - 40.
- [38] 焦春林, 高满屯, 史仪凯. 基于改进型聚类神经网络的图像分割[J]. 计算机工程与应用, 2007, 43(20): 93 - 95.
- [39] 薛岚燕, 郑胜林, 潘保昌, 等. 基于神经网络的灰度图像阈值分割方法[J]. 广东工业大学学报, 2005, 22(4): 67 - 72.

6 结束语

以上的改进方法用 VFP6.0 已进行了验证。关联规则的应用很广泛, 而它的经典算法 Apriori 算法中的频繁项集求解是耗时最多的工作, 那么提高了频繁项集的求解速度, 也就加快了关联规则的求解速度。

参考文献:

- [1] 陈文伟, 黄金才. 数据仓库与数据挖掘[M]. 北京: 人民邮电出版社, 2004.
- [2] Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large databases[C]//Proceedings of the ACM SIGMOD Conference on Management of data (ACM SIGMOD'93). Washington, USA: [s. n.], 1993: 207 - 216.
- [3] 袁万莲, 郑诚, 翟明清. 一种改进的 Apriori 算法[J]. 计算机技术与发展, 2008, 18(5): 52 - 53.
- [4] 何中胜, 庄燕滨. 基于 Apriori & Fp - growth 的频繁项集发现算法[J]. 计算机技术与发展, 2008, 18(7): 46 - 47.
- [5] 吴志丹, 赵大宇, 唐恒永. 一种改进的关联规则挖掘算法[J]. 沈阳师范大学学报: 自然科学版, 2006(3): 258 - 259.
- [6] Agrawal R, Srikant R. Fast Algorithms for Mining Association Rules in Large Database[C]//Proceeding of the 20th International Conference on Very Large Databases. Santiago, Chile: [s. n.], 1994: 487 - 499.
- [7] 胡吉明, 鲜学丰. 挖掘关联规则中 Apriori 算法的研究与改进[J]. 计算机技术与发展, 2006, 16(4): 99 - 101.
- [8] Cheung D W, Han J, Ng V, et al. A fast distributed Algorithm for mining association rules[C]//In: Proc 1996 Int Conf Parallel and Distributed Information Systems. Miami Beach, FL: [s. n.], 1996: 31 - 44.
- [9] 郭有强. 一种高效的关联规则维护算法研究与实现[J]. 计算机技术与发展, 2007, 17(10): 123 - 126.