

一种基于预判筛选的频繁项集挖掘算法

李德辰¹, 吕一帆¹, 赵学健²

(1. 南京邮电大学 物联网学院, 江苏 南京 210023;

2. 南京邮电大学 现代邮政学院, 江苏 南京 210003)

摘要:频繁项集挖掘作为关联规则挖掘技术的关键步骤,其性能对关联规则挖掘具有重要的意义。针对经典关联规则挖掘算法——Apriori 算法存在的产生候选项目集效率低和频繁扫描数据库等缺点,对 Apriori 算法的原理及效率进行分析,提出一种基于预判筛选策略的频繁项集挖掘算法。该算法通过对原始数据集的随机取样,得出样本频繁项集的支持度集合来进行预判筛选,从而对原始数据集候选项集进行二次剪枝,并且引入阻尼因子和补偿因子对预判筛选产生的误差进行修正,以保证算法的误判率和遗漏率。实验结果表明,该算法具有更好的时效性。

关键词:关联规则; Apriori; 数据挖掘; 预判筛选; 频繁项集

中图分类号: TP391

文献标识码: A

文章编号: 1673-629X(2018)05-0099-04

doi: 10.3969/j.issn.1673-629X.2018.05.023

A Frequent Item-set Mining Algorithm Based on Prejudgment and Screening

LI De-chen¹, LYU Yi-fan¹, ZHAO Xue-jian²

(1. School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing 210023, China;

2. School of Modern Post, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: Frequent item-set mining as a key step in mining association rules, its performance is of great significance to mining association rules. Aiming at the shortcomings of classical Apriori algorithm like low efficiency and frequent scanning database, we propose a frequent item-set mining algorithm based on prejudice and screening through analysis of the principle and efficiency of the Apriori algorithm. It obtains the support-set of frequent item-set for prejudgment and screening by random sampling of the original dataset, so as to make the second pruning of the candidate set from original dataset. The damping factor and the compensation factor are introduced to correct the error caused by the pre-selection screening to ensure the misjudgment rate and the omission rate of the algorithm. The experiments show that the proposed algorithm has better time efficiency.

Key words: association rules; Apriori; data mining; prejudging and screening; frequent item-set

0 引言

近年来,数据挖掘技术在各行各业的决策支持活动中扮演着越来越重要的角色^[1-2]。关联规则分析作为数据挖掘最活跃的研究领域之一,在精准营销^[3]、个性化医疗诊断^[4]、网络优化与管理^[5]等领域均有着广泛的应用。所谓关联规则就是隐藏在海量数据中的事物之间的联系和规律。在数据量急剧膨胀的今天,如何从海量数据中快速、高效地找出这些隐藏信息,提高关联规则分析算法的效率,具有十分重要的意义和应

用价值。关联规则挖掘通常分两步进行:频繁项集挖掘,即找出所有满足最小支持度的项集,找出的这些项集称为频繁项集;生成关联规则,在第一步产生的频繁项集的基础上生成满足最小置信度的规则,产生的规则称为强规则。频繁项集挖掘作为关联规则挖掘技术的关键步骤,其性能对关联规则挖掘具有重要的意义。

Agrawal 和 Skrikant 在 1994 年提出了第一个关联规则分析算法——Apriori 算法^[6]。Apriori 算法是最经典的关联规则分析算法之一。Apriori 算法使用重

收稿日期: 2017-04-24

修回日期: 2017-08-23

网络出版时间: 2018-02-07

基金项目: 国家自然科学基金(61373135, 61401225, 61572262, 61502246, 61672299); 中国博士后科学基金(2015M581844); 江苏省基础研究计划(自然科学基金)(BK20140883, BK20140894, BK20150869); 江苏省博士后科研资助计划项目(1501125B); 南京邮电大学校级科研基金(NY214101, NY215147)

作者简介: 李德辰(1997-), 男, 研究方向为关联规则分析; 赵学健, 博士, 副教授, 研究方向为数据挖掘、无线网络等。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20180207.1525.012.html>

复迭代的方法生成频繁项集。首先扫描数据库,得到所有项目的出现频率,并与给定的最小支持度阈值进行比较,得到频繁 1-项集 L_1 。接下来,对频繁 1-项集进行自连接,并根据频繁项集的向下闭包特性进行剪枝,产生候选频繁项集 2-项集 C_2 ,接下来进行扫描数据库判决,得到频繁 2-项集 L_2 。以此类推,直至得到所有频繁项集为止。

Apriori 算法在对候选频繁项集进行剪枝操作的过程中,用到了频繁项集的向下闭包特性。该特性是指如果一个集合是频繁项集,则它的所有子集都是频繁项集;反之,如果一个集合不是频繁项集,则它的所有超集都不是频繁项集。Apriori 算法利用频繁项集的向下闭包特性对候选频繁项集进行剪枝,从而有效地控制候选项集的指数增长。

从上述可以看出,Apriori 算法生成关联规则的过程包含两个步骤:挖掘隐藏在海量数据集中的所有频繁项集;根据挖掘出的频繁项集生成关联规则。其中第二步相对比较简单,第一步才是 Apriori 算法实现关联规则分析的关键,当然也是决定算法性能优劣的关键。目前对于 Apriori 算法的改进方法也大多数是针对第一步进行的。

Apriori 算法产生频繁项集的过程有两个重要特点。首先,Apriori 算法通过重复迭代生成频繁项集,在由候选频繁项集生成频繁项集的过程中,都要通过扫描数据库对候选频繁项集进行判别;其次,Apriori 算法在每次迭代过程中,都要通过自连接生成候选频繁项集。这两个特点使得算法虽然思想简单,较容易实现,但是却存在两个缺点:在规则产生过程中,算法必须反复扫描事务库,I/O 负载较大,且算法的运行效率较低;在自连接的过程中,会产生过多候选项集,使得挖掘的候选项集所含的项数过多,导致计算量惊人。这两个缺点使得 Apriori 算法在处理一些项集较多且长度较长的事务数据库时,显得力不从心。

为了克服 Apriori 算法存在的上述缺点,提出一种 A_RSPS 算法(Apriori with random sampling based pre-judgment and screening)。通过对原始数据集的随机取样,进行 Apriori 算法计算,得出样本频繁项集的支持度集合,再计算原始数据集的频繁项集,遍历数据之前通过之前得到的样本支持度集合进行预判筛选对候选项集进行二次剪枝,并且引入阻尼因子和补偿因子对预判筛选产生的误差进行修正,以减少扫描数据库的次数,降低算法的运算时间,提高算法的运算效率。

1 相关研究

研究人员对频繁项集挖掘算法进行了研究,取得了大量研究成果。文献[7]采用矩阵的方法表示数据

库,每个项目对应矩阵的一行,每个事务对应矩阵的一列,则矩阵的行向量之和为所对应项目在各事务中出现的次数,即该项目的支持度。可以看出,通过对矩阵的操作实现频繁项集的挖掘,无需多次扫描数据库,可以提高关联规则分析算法的时间效率,但是算法的空间复杂度较大。文献[8]使用 Hash 表存储事务数据以减少存储空间,同时使计算频繁项集更高效方便。此外,通过删除无用项表可以减少扫描 Hash 表的数量。用该方法在不损失频繁项集的前提下提高了发现频繁项集的效率。文献[9]对产生的每一个项集,采用包含两个线性表的类进行存储。事务标识符列表由支持该项集的所有事务标识符组成。因此一个项集的支持度就等于该项集的事务标识符列表长度。候选项集的支持度只要取其相应子集的事务标识符列表的交集得到,从而避免了为得到候选项集的支持度而去扫描数据库。文献[10]提出了一种新的产生候选集的方法,在 $k-1$ 项频繁集中的一个项集与其余所有项集进行连接,把连接得到的不同 k 项集存储,然后立即确定所有符合剪枝后的候选 k 项集。这样就省略了寻找 k 项集的所有 $k-1$ 项子集的费时剪枝操作,从而使剪枝步的平均扫描量大为减少。文献[11]把算法和负关联规则理论相结合,提出了一种基于负关联规则的数据挖掘算法。文献[12]提出的算法只需要一次数据库扫描。该算法在扫描数据库并计算每个项目的支持度时不会产生支持度为 0 的候选项,减少了候选项的数量。该文献还提到利用基于聚类的算法通过压缩事务数据库,通过节省无效的数据库扫描以提高算法的效率。文献[13]提出了基于用户的兴趣度的预处理的算法。该算法使用兴趣项排除不相关的项目以减少候选集 D ,其采用的纺织数据库包含众多参数,改进的算法只需要其中两个参数,同样减少了数据库扫描。文献[14]提出了一种有效的贪婪算法,以在给定的事务数据库中生成不相交的频繁项集的集合。该算法从给定的不相交频繁项集开始,发现更频繁的项目集。文献[15]提出预判筛选算法,该算法在 Apriori 算法连接、剪枝的基础上,添加了预判筛选的步骤,通过使用先验概率对候选频繁 k 项集集合进行缩减优化,并且引入阻尼因子和补偿因子对预判筛选产生的误差进行修正,以减少扫描数据库的次数,降低算法的运算时间,提高算法的运算效率。文中正是基于该文献提出的预判筛选的思想,结合采样思想进行的改进。

2 A_RSPS 算法

2.1 相关定义

假设 D 是挖掘的事务数据库,该数据库中包含 n 个事务,即 $D = \{T_1, T_2, \dots, T_n\}$ 。 I 为数据库中全部项

目的集合 $I = \{I_1, I_2, \dots, I_m\}$ 。对 $\forall T_q \in D$, 有 $T_q \subseteq I(1 \leq a \leq n)$ 。如果项目集 X 包含 k 个不同的项目, 称 X 为 k 项集。如果 $X \subseteq T_q$, 称项集出现在事务 T_q 中, 所有可能的 k 项集 X 组成集合 C_k 。统计该事件在 D 中发生的频率 P_x , 称为 X 在 D 中的支持度 (support), 给出一个 D 的最小支持度 min_support , 若 $P_x > \text{min_support}$, 则称 X 为频繁 k 项集, 所有可能的频繁 k 项集 X 组成集合 L_k 。

对于给定的事务数据库 D , 给定的最小支持度为 min_support 。 D 中客观存在的频繁项集集合为 L , 包含 N 个成员; 运行 ARSPS 算法所得频繁项目集集合为 L_a 。属于集合 L_a 但不属于集合 L 的项集数量记为 N_f , 属于集合 L 但不属于集合 L_a 的项集数量记为 N_o 。文中称属于集合 L_a 但不属于集合 L 的项集为误判项集, 其中误判率 $\text{MR} = N_f/N$, 称属于集合 L 但不属于集合 L_a 的项集为遗漏项集, 其中遗漏率 $\text{OR} = N_o/N$ 。

2.2 算法描述

ARSPS 算法寻找频繁项集的过程如下:

步骤 1: 对 D 进行随机取样取其子集 D_s , 取适当的 Δ_2 , 以 $(1 - \Delta_2) * \text{min_support}$ 对 D_s 进行 Apriori 算法运算构建频繁项集 L_s , 与对应的支持度集合 sample_support 组成一个筛选用的预判概率集合 $\text{PS_set}(L_s, \text{sample_support})$ 。

步骤 2: 扫描事务数据库 D , 对 D 中包含项目 I_i 的事务数 N_i 进行统计, 其中 $I_i \in I$, 得到候选 1 项集 $C_1 = I$, 及其支持度集合 $\text{support} = \{N_i/|D|, \in [1, m]\}$ 。

步骤 3: 对于 C_1 中的每一个候选项 C_i , 判断它是否存在于之前的先验概率集合 PS_set 中, 如果不在则把它从 C_1 中删去, 如果有, 取适当的 Δ_1 , 如果 C_i 大于 $\text{min_support} * (1 + \Delta_1)$ 那就把它添加到 L_1 , 并且从 C_1 中删除。最后扫描 C_1 , 删除那些 $N_i < \text{min_support}$ 的项集, 这样就得到了频繁 1 项集的集合 L_1 。

步骤 4: 假设 L_{k-1} 已生成, 现在可用它来生成 L_k , L_{k-1} 与自身进行连接得到候选 k 项集 $C_k, k \in \{2, 3, 4 \dots\}$, 第 1 次执行时 $k = 2$, 每循环执行一次 k 加 1。

连接过程如下: 对于 $\forall x_1, x_2 \in L_{k-1}$, 若 $x_1[1] = x_2[1], x_1[2] = x_2[2], x_1[k-2] = x_2[k-2], \dots, x_1[k-1] = x_2[k-1]$, 则将 x_1, x_2 连接生成候选项 $c = \{x_1[1], x_1[2], \dots, x_1[k-1], x_2[k-1]\}$ 。

步骤 5: 根据 Apriori 原理 (如果某个项集是频繁的, 那么它的所有子集也是频繁的), 从候选 k 项集 C_k 中删除所有 $k-1$ 项子集不完全包含在频繁 $k-1$ 项集 L_{k-1} 中的项。

步骤 6: 对于剪枝后的 C_k 中的每一个候选项 C_i , 判断它是否存在于之前的先验概率集合 PS_set 中, 如果不存在, 则把数据从 C_k 中删去, 如果存在且大于

$\text{min_support} * (1 + \Delta_1)$, 那就把它添加到 L_k , 并且从 C_k 中删除。

步骤 7: 扫描数据库, 判断预判筛选后的每个成员是否满足最小支持度要求, 满足则加入频繁项集循环执行直至为空, 不能发现更大的频繁项目集为止。

步骤 8: 最终获得的频繁项目集集合为 L 。

3 实验分析

采用 Python 语言实现了 Apriori 和改进的 ARSPS 算法, 并通过实验对两个算法进行了对比。数据集使用 Frequent Item-set Mining Dataset Repository (<http://fimi.ua.ac.be/data/>) 网站提供的 IBM Almaden Quest 研究组生成的数据, 算法增加的取样步骤中设置取事务数的一定百分比作为采样数据, 引入阻尼因子和补偿因子两个参数, 通过合理设置阻尼因子 1 和补偿因子 2 可有效降低误判率和遗漏率。

首先, 设计实验 1 对阻尼因子和补偿因子的取值进行分析, 每一组实验采用控制变量法, 相同参数重复实验 5 次取平均值。表 1 表示 $\text{min_support} = 0.02$, 阻尼因子 Δ_1 取值从 0.05 到 0.25 的过程中事务数分别为 5k, 10k, 25k, 50k 对应的频繁项集误判率。由表可知, 当同一大小数据集 Δ_1 取值变大时误判率逐渐减小, 当 Δ_1 取值确定时误判率随事务数增大而减小, 尤其当事务数大于 10k 后, Δ_1 大于 0.1 后发生误判的概率已经低于 1%。

表 1 阻尼因子-误判率

事务数	不同阻尼因子对应的误判率/%				
	0.05	0.1	0.15	0.2	0.25
5k	15.16	10.31	8.57	5.22	4.84
10k	9.67	6.41	4.58	1.96	1.57
25k	4.65	1.68	0.77	0.52	0.39
50k	4	1.2	0.39	0.13	0.13

实验 2 的数据如表 2 所示, 表示 $\text{min_support} = 0.02$, 补偿因子 Δ_2 取值从 0.05 到 0.2 变化过程中 5k, 10k, 25k, 50k 四组事务数据库的遗漏率。同实验 1 一样, 事务数越大, 遗漏率越小, Δ_2 越大, 遗漏率越小。尤其在事务数较小的情况下, Δ_2 取较小值则会造成较大的遗漏率, 而数据集很大时则遗漏率小于 1%, 可以接受。

表 2 补偿因子-遗漏率

事务数	不同补偿因子对应的遗漏率(%)			
	0.2	0.15	0.1	0.05
5k	4.35	8.94	9.69	16.27
10k	2.09	4.05	5.35	6.92
25k	0.52	1.42	2.58	3.87
50k	0.13	0.27	0.53	2.53

实验 3 对算法运行时间与事务数规模的关系进行了分析, 设置 $\text{min_support} = 0.02$, 在保证误判率和遗

漏率的情况下 Apriori 和改进的 A_RSPS 算法运行时间如图 1 所示。由图 1 可见,Apriori 算法的运行时间随着事务数增大迅速增加,100k 事务数的数据集需要约 193 s,而改进算法对于 100k 事务数的数据集需要约 34 s。可以看出,A_RSPS 相对于 Apriori 算法来说,时间效率得到了较大提升。

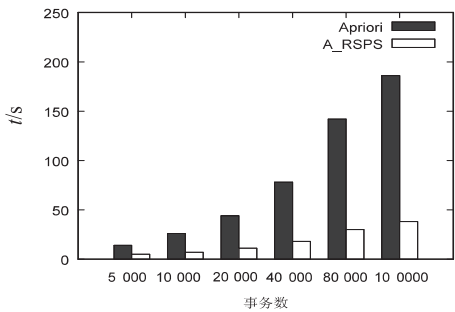


图 1 算法运行时间随事务数的变化

实验 4 对算法运行时间随最小支持度 min_support 的变化情况进行了分析。对于 10k 的数据集,在保证误判率和遗漏率的情况下,分别设置 min_support 为 0.01,0.02,0.04,0.08,为保证 min_support 取较小情况下的误判率不会过高,选择 10% 取样,设置 $\Delta_1=0.4,\Delta_2=0.35$,见表 3。

表 3 最小支持度对运行时间的影响

算法	不同最小支持度对应的运行时间/s			
	0.01	0.02	0.04	0.08
Apriori	62.42	10.85	1.73	1.55
A_RSPS	15.18	4.78	1.32	0.85

实验 5 对算法取样率对运行时间、遗漏率和误判率的影响进行分析。设定 min_support = 0.02,10k 数据集, $\Delta_1=0.25,\Delta_2=0.25$,分别取样 5%,10%,15%,20%,25%,在同一参数下进行 5 次测试取均值,相应的运行时间、遗漏率和误判率如表 4 所示。由该表可知,改进算法占原始算法时间比随着取样率的增加而增加,15% 取样率时约需要消耗 44% 原始 Apriori 算法所需的时间,同时,遗漏率和误判率相应减少,在 30% 取样率时已经几乎不出现遗漏和误判了,而取 10% 以下取样率时遗漏率和误判率会明显增大,不宜采用。

表 4 算法取样率对运行时间、遗漏率和误判率的影响

取样率	5%	10%	15%	20%	25%	30%
t_0/s	11.05	11.29	11.02	10.88	10.90	11.15
t_1/s	2.60	4.31	4.92	6.65	7.35	8.90
$t_1/t_0/\%$	23.52	38.15	44.60	61.15	67.45	79.79
遗漏率/%	4.71	1.44	1.31	0.13	0.26	0
误判率/%	5.49	2.09	0.87	0.78	0.13	0

4 结束语

文中提出一种基于预判筛选和采样思想的关联规

则挖掘算法 A_RSPS。该算法在对数据集处理之前取样部分数据进行经典 Apriori 算法计算得出样本数据的支持度,在原始算法连接、剪枝的基础上,增加了预判筛选的步骤,通过使用样本计算得到的支持度对候选频繁 k 项集集合进行缩减优化,从而减少关联规则挖掘过程中扫描数据库的次数。此外,算法引入阻尼因子和补偿因子对预判筛选引起的误判率和遗漏率进行控制。经实验验证,A_RSPS 算法在保证误判率和遗漏率的前提下降低了算法的运算时间,提高了算法的运算效率。

参考文献:

[1] 王光宏,蒋平.数据挖掘综述[J].同济大学学报:自然科学版,2004,32(2):246-252.

[2] 毕建欣,张岐山.关联规则挖掘算法综述[J].中国工程科学,2005,7(4):88-94.

[3] 阮利男.大数据时代精准营销在京东的应用研究[D].成都:电子科技大学,2016.

[4] 黄新霆,包小源,俞国培,等.医疗大数据驱动的个性化医疗服务引擎研究[J].中国数字医学,2014,9(8):5-7.

[5] 岳彦杰.基于规则的网络数据关联分析器的优化设计[D].哈尔滨:哈尔滨工业大学,2008.

[6] AGRAWAL R,SRIKANT R. Fast algorithms for mining association rules [C]//Proceedings of the 20th international conference on very large data bases. [s. l.]:[s. n.],1994:487-499.

[7] 马盈仓.挖掘关联规则中 Apriori 算法的改进[J].计算机应用与软件,2004,21(11):82-84.

[8] 陈文庆,许棠.关联规则挖掘 Apriori 算法的改进与实现[J].微机发展(现名:计算机技术与发展),2005,15(8):155-157.

[9] 刘华婷,郭仁祥,姜浩.关联规则挖掘 Apriori 算法的研究与改进[J].计算机应用与软件,2009,26(1):146-149.

[10] 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.

[11] 张玺.数据挖掘中关联规则算法的研究与改进[D].北京:北京邮电大学,2014.

[12] RAJESWARI K. Improved Apriori algorithm—a comparative study using different objective measures [J]. International Journal of Computer Science and Information Technologies, 2015,6(3):3185-3191.

[13] INGLE M G,SURYAVANSHI N Y. Association rule mining using improved Apriori algorithm[J]. International Journal of Computer Applications,2015,112(4):37-41.

[14] PALSHIKAR G K,KALE M S,APTE M M. Association rules mining using heavy itemset [C]//Proceedings of data & knowledge engineering. [s. l.]:[s. n.],2007.

[15] 赵学健,孙知信,袁源.基于预判筛选的高效关联规则挖掘算法[J].电子与信息学报,2016,38(7):1654-1659.