高激励项集的挖掘研究

余光柱[1,3],刘旭辉[2],邵世煌[1]
YU Guang-zhu[1,3],LIU Xu-hui[2],SHAO Shi-huang[1]

1.东华大学 信息科学与技术学院,上海 201600
2.长江大学 机械工程学院,湖北 荆州 434000
3.湖北警官学院 计算机系,武汉 430034
1.College of Information Science and Technology,Donghua University,Shanghai 201600,China
2.College of Mechanical Engineering,Yangtze University,Jingzhou,Hubei 434000,China
3.Department of Computer Science,Hubei University of Police,Wuhan 430034,China
E-mail:guang216@126.com

YU Guang-zhu,LIU Xu-hui,SHAO Shi-huang.Study of high motivation itemsets mining.Computer Engineering and Applications,2009,45(33):125-127.

Abstract: Algorithms for support-based association rules mining can only discover frequent itemsets,but can not discover the non-frequent itemsets with high utility values;Utility-based association rules mining aims at discovering high utility itemsets,without considering the itemsets whose utility values are not high but the product of the support and utility of the same itemset is very large.To solve the problem,a new measure is proposed,i.e.,motivation,to measure the importance of an itemset and a down-top algorithm called HM-Two-Phase-Miner to discover high motivation itemsets.Motivation integrates the advantages of support and utility,and thus can reflect both the semantic significance and statistical significance of an itemset.In HM-Two-Phase-Miner algorithm,transaction-weighted motivation downward closure property is adopted to cut down the search space.
Key words: high motivation itemset;association rule;support;utility-based

摘 要: 基于支持度的关联规则只能找出所有的频繁集,无法找到那些非频繁但效用很高的项集;基于效用的关联规则致力于发现所有高效用项集,无法找到效用不高但支持度与效用的积很大的项集。为克服支持度与效用的不足,提出了一种新的项集重要性的度量方法(即激励)及一种自下而上的挖掘高激励项集的算法 HM-Two-Phase-Miner。激励集成了支持度与效用的优点,能同时表达项集的语义特性与统计特性。HM-Two-Phase-Miner 利用事务权重激励向下封闭特性进行减枝,有效提高了算法的性能。
关键词: 高激励项集;关联规则;支持度;基于效用
DOI: 10.3778/j.issn.1002-8331.2009.33.041 **文章编号:** 1002-8331(2009)33-0125-03 **文献标识码:** A **中图分类号:** TP182

1 引言

基于支持度的关联规则及其改进算法[1-4]用支持度衡量用户的兴趣,能找出所有的频繁集,却无法找到那些非频繁但效用很高的项集,导致有用知识的丢失。例如,在事务数据库中,有些项集的支持度不高,但能给商家带来很多利润,更能引起商家的兴趣。基于效用的关联规则(UBARM)[5-7]用效用代替支持度评价项集的重要性,致力于发现所有高效用项集。但是,基于效用的关联规则不能发现效用不高但发生比较频繁、支持度与效用值的积(后面定义为激励)很大的项集。虽然这些项集的效用值不高(略低于用户定义的阀值),但很可能引起用户的兴趣。相对于那些效用很高但支持度很低的项集,这些激励很大的项集往往意味着一个稳妥可靠的决策方案:决策成功后带来的效益不是很高,但成功的可能性很大。现实生活中,多数人还是更青睐这种稳妥的方案,而对成攻后效益很大但成功率很低的方案(如买体育彩票)持谨慎态度。

造成用户感兴趣模式丢失的重要原因在于,在度量用户兴趣这个问题上,前述的两种关联规则都做了过于简单的假设:基于支持度的关联规则假设用户只对频繁发生的项集感兴趣,基于效用的关联规则假设用户只对高效用项集感兴趣。事实上,决定人们兴趣的因素既有主观方面的,又有客观方面的。支持度作为一种客观度量,不能反映项集的语义特性;效用作为一种主观度量,不能反映项集的统计特性[8]。因此,单纯的基于效用或基于支持度的关联规则都无法准确表示用户的兴趣。

期望理论[9]认为,激励是评价、选择的过程,人们采取某项

基金项目: 国家教育部博士点基金(the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No. 20060255006)。
作者简介: 余光柱(1969-),男,博士研究生,研究方向为数据挖掘,智能控制;刘旭辉(1966-),男,博士研究生,研究方向为数据挖掘;邵世煌(1938-),男,教授,博士生导师,研究方向为智能控制、软计算、数据挖掘等。
收稿日期: 2008-07-02 **修回日期:** 2008-08-03

行动的动力(或激励力)取决于其对行动结果的价值评价(效价)和预期实现目标可能性的估计(期望值)。换言之,激励力的大小取决于效价与期望值的乘积:

激励力(motivation)=效价(valence)×期望值(expectancy) (1)

因此,对于一个项集,决定人们兴趣的,至少应包括项集的支持度(与期望值对应)与效用(与效价对应)。根据式(1),用支持度与效用的积(激励)来反映项集的统计特性和语义特性,并提出了一种高激励项集的挖掘算法。

2 概念与定义

设 $I=\{i_1,i_2,\cdots,i_m\}$为项目集合,$T=\{t_1,t_2,\cdots,t_n\}$为事务数据库。每一事务 $t_q(t_q\in T)$是 I 的一个子集,即 $t_q\subseteq I$。对于 I 中的一个子集 S,有 $S\subseteq t_q$,就说 t_q 包含 S。为便于描述,参考有关文献中的概念[5-7],作如下定义:

定义 1 项集 S 的事务集,记为 T_s,是所有包含 S 的事务的集合,即

$$T_s=\{t_q|S\subseteq t_q,t_q\in T\} \quad (2)$$

显然,如果 $S_1\supseteq S_2$,$T_{S_1}\subseteq T_{S_2}$。

定义 2 项目 i_p 在事务 t_q 中的效用(项目的事务效用),记为 $l(i_p,t_q)$,指事务 t_q 发生时项目 i_p 带给用户的效用。为便于理解,假设所指效用为经济效用。在事务数据库中,项目的事务效用是该项目的单位利润与销售量之积。

定义 3 项集 S 在事务 t_q 中的效用(项集的事务效用),记为 $l(S,t_q)$,是 S 中所有项目 i_p 的事务效用的和。即

$$l(S,t_q)=\sum_{i_p\in S}l(i_p,t_q) \quad (3)$$

当 $S=t_q$ 时,简称事务的效用,记为 $l(t_q,t_q)=\sum_{i_p\in t_q}l(i_p,t_q)$。显然,根据定义,式(4)成立:

$$l(S,t_q)\leqslant l(t_q,t_q)(S\subseteq t_q) \quad (4)$$

定义 4 项集 S 的效用,记为 $u(S)$,指项集 S 在所有事务 t_q 中的效用的和,即

$$u(S)=\sum_{t_q\in T_s}l(S,t_q) \quad (5)$$

定义 5 项集 S 的激励,记为 $m(S)$,指项集 S 的支持度 $s(S)$与效用值的积,即

$$m(S)=s(S)\times u(S) \quad (6)$$

如果项集的激励不小于用户定义的阈值 min*motivation*,此项集是高激励项集。否则,是低激励项集。目标就是要找出所有的高激励项集。

定义 6 项集 S 的事务权重效用[6](transaction-weighted utilization),记为 $twu(S)$,是所有包涵 S 的事务的效用的和,即

$$twu(S)=\sum_{t_q\in T_s}l(t_q,t_q) \quad (7)$$

如果项集的事务权重效用 $twu(S)$不小于用户定义的阈值 TWmin*util*,此项集是高事务权重效用项集。否则,是低事务权重效用项集。显然,$twu(S)\geqslant u(S)$。

定义 7 项集 S 的事务权重激励,记为 $twm(S)$,是 S 的事务权重效用与它的支持度的积。即

$$twm(S)=twu(S)\times s(S) \quad (8)$$

如果一个项集 S 的事务权重激励 $twm(S)$不小于用户指定的阈值 TWmin*motivation*,则 S 是一个高事务权重激励项集。

3 相关研究

沈一栋等曾提出过一个面向目标的基于效用的关联规则挖掘模型(OOA Model)[3]。OOA 模型同时用支持度和效用度量项集的重要性,能发现数据集中高效用频繁集。但是,OOA 模型及他提出的 OOApriori 算法与该文的目标有下列不同:(1)OOA 关联规则不要求项集的支持度与效用值的乘积大于等于某一阈值;(2)OOA 模型中支持度阈值 min*sup* 必须设置得比较大,否则会产生太多的频繁集。因此,OOA 模型仍然会丢失一些支持度不高,但激励很大的模式。在高激励项集挖掘中,算法通过激励阈值 min*motivation* 来除去不太重要的规则,减枝时不涉及支持度阈值 min*sup* 和效用阈值 min*util*。但是,在实验中,参考 min*sup* 和 min*util* 的值决定 min*motivation* 的大小。事实上,支持度阈值 min*sup* 和效用阈值 min*util* 往往可以设置得很小,因为同时满足 min*sup* 和 min*util* 的项集很少(见图 1、2)。当然,算法中也可以利用 min*sup* 和 min*util* 阈值减枝,过滤掉一些非常频繁但效用太低或虽然效用很高但属于偶然的无用模式(项集),即使这些项集的激励大于激励阈值 min*motivation*。

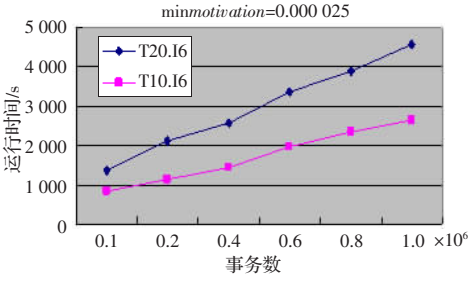

图 1 事务数变化对算法性能的影响

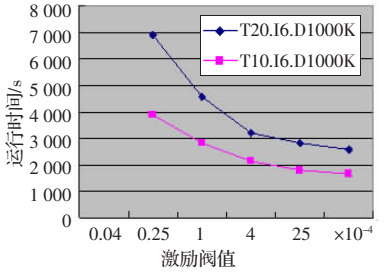

图 2 激励的变化对算法性能的影响

文献[10]提出用"一般效用(general utility)"度量项集的重要性。按定义,项集 S 的一般效用 $gu(S)$等于它的支持度与效用的加权和,即 $gu(S)=\lambda s(S)+(1-\lambda)u(S)$。"一般效用"的确同时反映了项集的语义特性与统计特性,但权值 λ 的确定带有较大的随意性,概念的意义也不如"激励"直观。激励是基于概率论与管理学的,更好理解。

文献[6]提出了一种基于效用的关联规则挖掘算法 Two-phase,与其他效用挖掘算法如一样,Two-phase 会丢失一些高激励项集。但是,它提出的事务权重效用向下封闭特性为该文的研究奠定了基础。该文所证明的事务权重激励向下封闭特性其实是事务权重效用向下封闭特性的扩展。文献[10]对 Two-phase 进行了必要修改,利用一般事务权重效用向下封闭的特性(general transaction-weighted downward closure property)减枝。

4 算法

4.1 激励约束的特性

有关研究表明[3,5],效用约束既不是单调的(monotone)、非

单调的(anti-monotone)、可转换的(convertible),也不是简洁的(succinct)。根据激励的定义可知,激励约束既不是单调的、非单调的、可转换的,也不是简洁的。

定理 1(事务权重效用向下封闭特性) 设 S^k 是一 k-项集,S^{k-1} 是一$(k-1)$-项集,且 $S^{k-1}\subset S^k$。如果 S^k 是一高事务权重效用项集,那么,S^{k-1} 也是一高事务权重效用项集。

证明 设 T_{S^k} 是所有的包涵项目集 S^k 的事务的集合,$T_{S^{k-1}}$ 是所有的包涵项目集 S^{k-1} 的事务的集合。因为 $S^{k-1}\subset S^k$,那么,$T_{S^{k-1}}$ 是 T_{S^k} 的一个超集。根据定义 6(式 7),有

$$twu(S^{k-1})=\sum_{t_q\in T_{S^{k-1}}} l(t_q,t_q)\geqslant \sum_{t_q\in T_{S^k}} l(t_q,t_q)=twu(S^k)\geqslant TW\ \min util$$

证毕。

定理 2(事务权重激励向下封闭特性) 设 S^k 是一 k-项集,S^{k-1} 是一$(k-1)$-项集,且 $S^{k-1}\subset S^k$。如果 S^k 是一高事务权重激励项集,那么,S^{k-1} 也是一高事务权重激励项集。

证明 根据定理 1 可知,$twu(S^{k-1})\geqslant twu(S^k)$。又由于 $s(S^{k-1})\geqslant s(S^k)$,有

$$twu(S^{k-1})\times s(S^{k-1})\geqslant twu(S^k)\times s(S) \tag{9}$$

如果 $twu(S^k)\times s(S)\geqslant TW\ \min motivation$,则 $twu(S^{k-1})\times s(S^{k-1})\geqslant TW\ \min motivation$ 成立。

证毕。

定理 3 设 HTWM 为数据库 T 中所有高事务权重激励项集的集合,HM 为 T 中所有高激励项集的集合。如果 $TW\min motivation$ 等于 $\min motivation$,则 $HM\subseteq HTWM$。

证明 $\forall S\in HM$,如果 S 是一高激励项集,那么有:

$$TW\ \min motivation=\min motivation\leqslant s(S)\times u(S)=$$

$$s(S)\times\sum_{t_q\in T_S} l(S,t_q)\leqslant s(S)\times\sum_{t_q\in T_S} l(t_q,t_q)=s(S)\times twu(S)=twm(S)$$

证毕。

这样,通过设 $TW\ \min motivation=\min motivation$,根据定理 3,可利用事务权重激励向下封闭特性进行减枝,压缩搜索空间。

4.2 算法

基于上述的减枝策略,提出了一种新的类似于 Two-Phase 的算法,称为 HM-Two-Phase-Miner。HM-Two-Phase-Miner 算法采用了自底而上的搜索策略,反复从$(k-1)$-项集生成 k-项集,并计算候选集的激励。程序描述如下。

算法 HM-Two-Phase-Miner

输入 数据库 T,阈值 minmotivation

输出 高激励项集集合 HM

```
1.  {
2.  C_k^HTWM=ϕ; C^HTWM=ϕ; //C_k^HTWM 为高事务权重激励 k-项集的候选集,k 为项集大小;C^HTWM 为高事务权重激励项集的候选集。l 为高事务权重激励项集的最大长度。
3.  HM=ϕ;  //高激励项集的集合
4.  k=1;
5.  扫描数据库 T,得到 C_1^HTWM;
6.  While (|C_k^HTWM|>0)
7.  {
8.      k=k+1;
9.      C_k^HTWM=Generate(C_{k-1}^HTWM);
10.     C^HTWM=C_k^HTWM ∪ C^HTWM;
11.     C_k^HTWM=CalculateAndDiscoverHTWM(C_k^HTWM,T,TW minmotivation);
12. }
13. HM=HM∪CalculateAndDiscoverHM(C^HTWM,T,minmotivation);
14. Return HM;
15. }
```

HM-Two-Phase-Miner 算法的第 1 步至第 4 步是初始化。第 5 步扫描数据库,得到高事务权重激励项集的候选集 C_1^{HTWM}。第 7 至第 12 步循环扫描数据库,生成并检验不同长度的候选集。其中,第 9 步 Generate 函数通过 C_{k-1}^{HTWM} 内$(k-1)$-项集的连接运算,产生高事务权重激励项集(k-项集)的候选集。第 10 步把各种长度的候选项集加入到 C^{HTWM}。第 11 步 CalculateAndDiscoverHTWM 函数计算并发现 C_k^{HTWM} 中高事务权重激励 k-项集,作为下一步生成 C_{k+1}^{HTWM} 的候选。第 9 步生成的 C_k^{HTWM} 包含了所有的高激励 k-项集,但也可能包含事务权重激励不高的 k-项集。为缩小搜索空间,应尽早去掉这些事务权重激励不高的 k-项集。在得到 C^{HTWM} 后,第 13 步再次扫描数据库,计算 C^{HTWM} 中项集的真实激励,得到高激励项集。

HM-Two-Phase-Miner 和 Two-Phase 的结构与 Apriori 相似,不同之处有:(1)减枝策略不同。Apriori 利用频繁集向下封闭的特性减枝,Two-Phase 利用项集的事务权重效用向下封闭特性减枝,而 HM-Two-Phase-Miner 利用项集的事务权重激励向下封闭特性进行减枝;(2)在生成 k-项集的候选集时,Apriori 仅通过大的$(k-1)$-项集(L_{k-1})的连接运算就可产生 C_k,而 HM-Two-Phase-Miner 和 Two-Phase 则通过候选集内的$(k-1)$-项集的连接运算产生新的候选集。换句话说,通过老候选集产生新候选集;(3)相对于 Apriori,HM-Two-Phase-Miner 和 Two-Phase 需要多扫描数据库一次,以计算候选项集的真实激励。这加重计算复杂性,但实验表明,由于减枝策略有效,算法性能良好。

5 实验与分析

实验在浪潮 XEON 服务器上进行。CPU 主频 2.4 G,内存 4 G,运行 windows 2003,程序用 Delphi 7 编写。实验用的数据集为 T10.I6.D1000K 和 T20.I6.D1000K,项目数为 1 k,由 IBM 的数据发生器生成(2007/5/16.http://www.almaden.ibm.com/cs/projects/iis/hdb/Projects/data_mining/datasets/syndata.html)。数据集中只有 0 和 1,分别代表某项目是否出现在事务中,没有效用值。因此,实验中用 delphi 随机函数"RandG"产生随机值(高斯分布)来模拟事务中各项目的单位效用,用事务编号的模 100(TID MOD 100)来表达销售数量。这样,某一事务中某一项目的效用就等于项目的销售数量乘以该项目的单位效用。显然,各项目的单位效用是随机的,而销售量是固定的。这决定了每次挖掘的结果不同。

实验中,为便于理解激励阈值的意义与来历,可假设 $\min util=\min sup$。例如,$\min motivation=0.002\,5$ 意味着 $\min util=\min sup=0.05$,$\min motivation=0.000\,025$ 意味着 $\min util=\min sup=0.005$。事实上,支持度与效用都大于 0.05 的项集很少,所以对 minmoti-

(下转 134 页)