一种基于决策树的快速关联规则挖掘算法

陈雪飞
(广东财经职业学院信息管理系　广州 510420)

摘　要　本文对关联规则的挖掘问题进行了深入研究。在总结现有算法优缺点的基础上，提出了一种新的基于决策树的快速关联规则挖掘算法(RABDT)，结合决策树的构造过程，给出了算法的原理和实现步骤，并通过实验对比验证了算法的有效性。

关键词　数据挖掘，关联规则，决策树，算法

Rapid Association Rules Mining Algorithm Based on Decision Tree

CHEN Xue-fei
(Guangdong Vocational College Financial Information Management Department, Guangzhou 510420, China)

Abstract　In this paper, the issue of mining association rules is discussed deeply. A new mining algorithm, Rapid Rules Mining Algorithm based on Decision Tree, RABDT is proposed, after summarizing the advantage and disadvantage of current mining association rules algorithms. The principle and steps of RABDT are given with constructing a decision tree, and a contrast test shows that RABDT is far rapid and effective.

Keywords　Ddata mining, Association rules, Decision tree, Algorithm

数据挖掘(也有人称之为数据库中的知识发现)已经被认为是现代计算机应用中一个极富应用前景的新领域。在数据库中挖掘关联规则是数据挖掘领域中一个非常重要的研究课题，它是由 R. Agrawal 等人首先提出的，最初用于对大型零售商场的日交易数据进行统计分析，结果发现了一些出人意料的规律，按照此规律对商场内货品的摆放位置进行重新调整后，商场的营业收入得到了显著的提高。这是现代计算机技术对经济发展具有促进作用的最直接体现，因此有人将利用计算机软硬件技术促进经济发展这一新的研究领域称之为商业智能(Business Intelligence, BI)。

1　关联规则挖掘的一般形式

在常用的关系型数据库中，关联规则的挖掘问题可形式化描述如下：设 $I=\{I_1, I_2, \cdots I_m\}$ 是 m 个不同项目的集合，D 是针对 I 的实例的集合，每一个实例包含若干项目 $I_i, I_j, \cdots, I_k \in I$。关联规则表示为 $X \Rightarrow Y$，其中 $X \subset I, Y \subset I$，并且 $X \cap Y = \phi$。X 称作规则的前提，Y 是结果。一般把一些项目的集合称作项目集(itemset)。在项目集中项目的数量称为项目集的长度。关联规则 $X \Rightarrow Y$ 成立的条件是：①它具有支持度 s，即实例数据库 D 中至少有 $s\%$ 的实例包含 X 和 Y；②它具有置信度 c，即在实例数据库 D 中包含 X 的实例至少有 $c\%$ 同时也包含 Y。关联规则的挖掘问题就是在实例数据库 D 中找出具有用户给定的最小支持度 min_sup 和最小置信度 min_conf 的关联规则。因此，该问题可以分解成如下两个子问题：①找出存在于实例数据库中的所有频繁项目集。若项目集 X 的支持度 support(X) 不小于用户给定的最小支持度 min_sup，则称 X 为频繁项目集；否则，为非频繁项目集。②利用频繁项目集生成关联规则。对于每个频繁项目集 A，若 $B \subset A, B \neq \phi$，且 support(A)/support(B) ≥ min_conf，则有关联规则 $B \Rightarrow (A-B)$。由于子问题②相对来说实现较为容易，因此目前关联规则挖掘领域的研究重点多集中在子问题①上。

2　现有挖掘算法简述

在现有关联规则挖掘算法的研究中，最有代表性的是 Apriori 算法。Apriori 算法需要产生大量的候选项目集，并多次扫描数据库，扫描次数等于最大频繁项目集的长度，效率较低。随后很多学者在 Apriori 算法的基础上进行了研究，提出了若干改进算法，其中 Han 等人提出的 FP-growth 算法[1]是一种本质上不同于 Apriori 算法的高效算法，它不需要产生候选项目集，只需要两次扫描数据库。第一次扫描数据库得到频繁项目集 1；第二次扫描数据库，利用频繁项目集 1 过滤数据库中的非频繁项目，同时生成 FP 树。由于 FP 树涵盖了所有的频繁项目集，其后的频繁项目集的挖掘只需要在 FP 树上进行。FP-growth 算法的基本操作是计数和建立 FP 树。对 FP 树方法的性能研究表明：对于挖掘长的和短的频繁项目集，它是有效的和可伸缩的，并且比 Apriori 算法快大约一个数量级。但是 FP-growth 算法仍然有不可克服的缺陷：一是随着数据量的增加和数据复杂性的提高，FP 树的分支数将大大增加，管理的难度将越来越大；二是两次扫描数据库，需要较大的存储空间作为中转，对计算机硬件尤其是存储容量方面提出了较高要求。此外，该方法还存在着无法处理数据的缺值问题等弊端。

在决策树算法方面，比较有代表性的是 Quinlan 提出的 ID3 算法[2]和 C4.5 算法[3]。ID3 算法是基于信息熵的决策树分类算法，根据属性集的取值选择实例的类别。ID3 是一个典型的决策树学习系统，它以信息熵作为分离目标评价函数，采用自顶向下不可返回的策略，搜出全部空间的一部分，确保决策树最简单且每次需要的测试数据最少。ID3 算法构

陈雪飞　硕士，副教授，主要研究方向为信息安全、数据库等。

造的决策树平均深度较小，分类速度较快。但也有不足之处，如偏向于选择属性较多的属性，而属性较多的属性却不总是最优的属性，另外该算法学习简单的逻辑表达能力也较差。C4.5 算法是 ID3 算法的改进，继承了 ID3 的优点，是一种归纳学习算法。先从所有的事例中选取一部分构造决策树，再用剩下的事例测试决策树并对它进行调整。它不仅能处理连续值类型的属性，还可以对属性的取值集合进行等价类划分，划分在同一类的属性值在属性值判断时走向同一分支。C4.5算法不足之处在于采用分而治之的策略所得到决策树不一定是最优的，而且用该算法构造的决策树结构调整、性能改善较困难。

此外，还有其他研究学者从不同角度对这两种经典的决策树算法进行了改进，如选择新属性的启发式函数，或者利用遗传算法等，但基本都是沿用了这两种算法的核心思想。这两种算法的思想在扩充决策树属性的取值范围及改进选择分离属性等方面固然有优势，但决策树构造效率不高、削减数据库遍历次数多、I/O 操作频繁等也是突出问题，影响了算法的应用范围。本文创造性的将决策树方法引入关联规则挖掘中，提出了一种基于决策树的关联规则挖掘算法（rapid rules mining algorithm based on decision tree，RABDT）。它具有以下优点：①只需要一次扫描数据库，比 FP-growth 算法还少，使算法的速度更快；②不产生候选项目集，从而节约了内存空间，降低了对计算机硬件的要求。

3 基于决策树的快速关联规则挖掘算法——RABDT

决策树也是数据挖掘研究领域的一种有效工具，该方法是利用信息论中的信息增益寻找示例数据库中具有最大信息量的属性字段，建立决策树的一个节点，再根据该属性字段的不同取值建立树的分支；在每个分枝集中重复建立树的下一个节点和分支的过程。首先选取部分数据，按照广度优先的原则建立决策树，直到每个叶节点包括相同的类标记为止。然后利用剩余数据检验决策树，根据情况对决策树进行调整，直到建立一棵正确的决策树。从根节点到叶节点的一条路径就对应着一条规则，整棵决策树就对应着一组析取表达式规则。决策树方法的最大特点是直观，这与关联规则的表示形式有异曲同工之妙，也为结合关联规则提取算法创造了条件。

表 1 数据库实例及项目

实例	所含项目
T1	I_1, I_2, I_5
T2	I_1, I_2, I_3
T3	I_2, I_3
T4	I_1, I_2, I_3, I_5
T5	I_1, I_2, I_4
T6	I_2, I_3
T7	I_1, I_3
T8	I_2, I_4

3.1 决策树的定义及构造方法

设 $I=\{I_1, I_2, \cdots, I_m\}$ 是 m 个不同项目的集合。设任务相关的数据 D 是数据库实例的集合，其中每个实例 T 是项目的集合，使得 $T \subseteq I$。每一个实例有一个标识符，称作 T_i。设 X 是一个项目集，实例 T 包含 X 当且仅当 $X \subseteq T$。关联规则是形如 $X \Rightarrow Y$ 的蕴涵式，其中 $X \subset I, Y \subset I$，并且 $X \cap Y = \phi$。规则 $X \Rightarrow Y$ 在实例集 D 中成立的条件是：具有支持度 s，其中 s 是 D 中实例包含 XY（即 X 和 Y 二者）的百分比，它是概率 $P(XY)$；具有置信度 c，c 是 D 中包含 X 的实例同时也包含 Y 的百分比，它是条件概率 $P(Y/X)$。设序列 $S=\{I_1, I_2, \cdots, I_n\}$，将序列分成任意的两段 $S_1=\{I_1, I_2, \cdots, I_k\}$，$S_2=\{I_{k+1}, I_{k+2}, \cdots, I_n\}$，其中 $1 \leqslant k < n$，S_1，S_2 非空，则称 S_1 为 S_2 的前缀，S_2 为 S_1 的后缀。

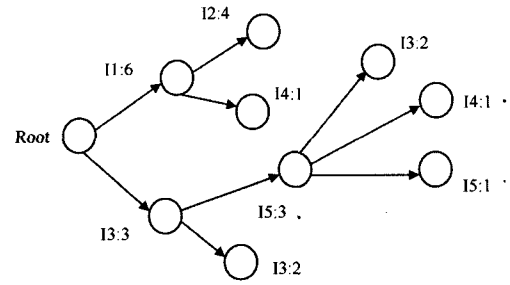

图 1 构造 D-tree 示意图（部分）

我们规定，决策树（Decision tree，简称 D-tree）有且只有一个根节点。设根节点为第 0 层，根节点的全部子节点组成第 1 层，子节点的全部子节点组成第 2 层，依次类推，直到叶子节点。当层数大于 1 时，每个子节点都有一个唯一的父节点。叶子节点所处的层数称为路径长度 L。节点索引表是由项目集 I 中的每一个项目 I_k 及支持度计数以及指向决策树中 I_k 的节点链指针组成。

下面通过一个例子来说明决策树的生成。设有如表 1 所示的实例数据库，数据库中有 9 个实例。假定实例中的项按字典次序存放。T_i 为实例，I_1 到 I_5 为该实例中的项目，构造出的决策树如图 1 所示。限于篇幅，此图只表示出了部分决策树。

决策树的构造方法如下：首先，创建树的根节点，用“Root”标记，并建立节点索引表的空表。然后，逐行扫描实例数据库，每个实例中的项目按 $I_1, I_2, I_3, \cdots, I_n$ 的次序排列，对每个实例创建一个分枝，并更新节点索引表中的计数。直到扫描完整个数据库，产生决策树和节点索引表。

以表 1 中的数据为例。首先，创建树的“Root”根节点和节点索引表空表。然后，依次扫描表中的实例。第一个实例包含项目$\{I_1, I_2, I_5\}$，因此，先给节点索引表中的项目 I_1，I_2 和 I_5 的支持度计数分别加 1，再构造树的第一个分枝$\{(I_1:1), (I_2:1), (I_5:1)\}$。该分枝具有 3 个节点，其中 Il 作为根的子女节点，I_2 链接到 I_1，I_5 链接到 I_2。

第二个实例包含项目 I_1，I_2 和 I_4，因此，先给节点索引表中的项目 I_1，I_2 和 I_4 的支持度计数分别加 1，再构造一个分枝$\{(I_1:1), (I_2:1), (I_4:1)\}$，该分枝具有 3 个节点，其中 I_2 作为根的子女节点，I_2 链接到 I_1，I_4 链接到 I_2。

第三个实例包含项目 I_2 和 I_3，因此，先给节点索引表中的项目 I_2 和 I_3 的支持度计数分别加 1，再构造一个分枝，该分枝与第二个实例已存在的路径共享前缀$\{I_2:1\}$，这样，我们将节点 I_2 的计数增加 1，并创建一个新节点$(I_3:1)$，它作为$(I_2:2)$的子女节点。

其余类推，直到所有实例完成。表 1 中数据生成的节点索引表和决策树如图 1 所示。这样，实例数据库频繁项目集的挖掘问题就转换成了挖掘决策树问题。而且，从图 1 我们可以发现，如果从叶子节点到根节点的方向逆向拆解决策树，

可以将决策树还原成原来的实例数据库，即决策树与实例数据库是可逆的。

3.2 挖掘决策树中频繁项目集的算法

从以上构造决策树的过程可以看出，最大频繁集的长度小于等于最长路径的长度 L。设路径为 $I_{j1}:M_1, I_{j2}:M_2, I_{jn}:M_n$，由于构造 T-tree 时，实例中的项目是按照序号排序的，且从根节点开始向下构造。因此同一条路径中的上层节点的数目一定大于或等于下层节点的数目。即一定有 $M_1>M_2>\cdots>M_n$，该路径的支持度计数为 $\min(M_1, M_2, \cdots, M_n)=M_n$。而且，由于各个路径具有相同的后缀，则各个路径的交集必包含该后缀，因此路径的支持度计数为其后缀的支持度计数，而各路径的后缀部分是互不相交的，因此交集的支持度计数等于后缀的支持度计数之和，即具有相同后缀全部路径的交集 A 的支持度计数等于各个路径的子集 A 支持度计数之和。通过以上分析，基于决策树的快速关联规则挖掘(RABDT)算法可以描述如下：

输入：决策树 D-tree
输出：所有的频繁项目集和支持度计数
step1：if Tree 含单个路径 Pthen
step2：　for 路径 P 中节点的每个组合(记作 β)
step3：　　产生模式 $\beta \cup \alpha$，其支持度为 β 中节点的最小支持度；
step4：else for each α_i 在 Tree 的头部{
step5：产生一个模式 $\beta=\alpha_i \cup \alpha$，其支持度 $=\alpha_i$.support；
step6：构造 β 的条件模式基，然后构造 β 的条件决策树 Treeβ；
step7：if Tree$\beta \neq \phi$ then
step8：调用 T_growth (Treeβ, β)；}

用上述算法挖掘决策树图 1，设支持度阈值为 2/9(即最小支持度计数为 2)。挖掘过程简述如下：先考虑 I_5，I_5 出现在决策树的两个分枝。这些路径由分枝 $(I_1\ I_2\ I_5:1)$ 和 $(I_1 I_2 I_3 I_5:1)$ 形成。考虑 I_5 为后缀，它的两个对应前缀路径是 $(I_1 I_2:1)$ 和 $(I_1 I_2 I_3:1)$，它们形成 I_5 的条件模式基，它的条件决策树只包含单个路径 $\{I_1:2, I_2:2\}$，不包含 I_3，因为它的支持度小于最小支持度计数。该单个路径产生频繁项目集的所有组合：$(I_1 I_5:2)$，$(I_2\ I_5:2)$，$(I_1\ I_2 I_5:2)$。

与以上分析类似，I_4 的两个前缀形成条件模式基 $(I_1 I_2:1)$ 和 $(I_2:1)$，产生一个单节点的条件决策树 $\{I_2:2\}$，导出一个频繁项目集 $(I_2\ I_4:2)$。I_3 的条件模式基是 $(I_1\ I_2:2)$，$(I_1:2)$ 和 $(I_2:2)$，它的条件决策树有两个分枝 $\{I_1:4, I_2:2\}$ 和 $\{I_2:2\}$，产生频繁项目集 $(I_1 I_3:4)$，$(I_2 I_3:4)$，$(I_1 I_2 I_3:2)$。I_2 的条件模式基是 $(I_1:4)$，产生一个单节点的条件决策树 $\{I_1:4\}$，导出一个频繁项目集 $(I_1 I_2:4)$。

3.3 实验及结果对比

为了验证 RABDT 算法的实际效果，我们进行了实验，并与 FP-growth 算法的执行时间进行了比较，实验数据由关联规则数据合成程序产生，最大项目数为 10，平均每条实例的长度为 5，生成的 4 个实例数据库名称、实例数目以及两种算法的执行时间如表 2 所示。

表 2　实验数据及实验结果

实验数据	实例数目	FP-growth 算法执行时(单位：秒)	RABDT 算法执行时间(单位：秒)
Testl.dat	312	0.28	0.12
Test2.dat	4929	2.48	1.61
Test3.dat	42286	18.79	9.15
Test4.dat	124201	136.55	22.60

从实验结果来看，RABDT 算法的速度明显比 FP-growth 算法要快，而且随着数据量的增加，该算法的优势越明显，运算速度提高的倍数越多。

结束语　采用决策树方法挖掘关联规则有几个优点，一是决策树方法结构简单，可以生成便于人们理解的规则；二是决策树模型效率较高，计算量相对来说不是很大，对训练集数据量较大的情况较为适合；第三，决策树方法通常不需要额外的领域知识，自组织性较强。本文提出的 RABDT 算法只需要一次扫描实例数据库，可以极大地提高搜索效率，而且内存占用仅仅和实例数据库中项目的数量有关，而与实例数据库本身的大小无关，因此该算法可以挖掘海量数据库，具有良好的可伸缩性。RABDT 算法也可以实现增量挖掘，因为决策树和实例数据库是互逆的，可以将决策树保存为文件。当实例数据库的内容增加后再进行关联规则挖掘时，只要将决策树读入内存，然后对实例数据库的增量部分进行挖掘即可。当然，本文选取的实验数据大多是离散数据，对有时间顺序的连续数据的处理能力，还需要进一步加以实验研究。

参 考 文 献

[1] Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation// Proceedings of the 2000 ACM-SIGMOD International Conference on Management of Data. Dallas, TX, 2000: 1-12

[2] Quinlan J R. Induction of decision trees. Machine Learning, 1986, 1 (1): 81-106

[3] Quinlan J R. C4.5: Programs for machine learning. San Mateo: Morgan Kaufmann Publisher, 1993: 170-247

[4] 萨师煊，王珊. 数据库系统概论(第三版). 北京：高等教育出版社，2000

[5] 毛国君，刘椿年. 基于项目序列集操作的关联规则挖掘算法. 计算机学报，2002，25(4)：417-422

[6] 刘红岩，陆宏钧，陈剑. 利用数据库技术实现的可扩展的分类算法. 软件学报，2002，13(6)：1076-1081

(上接第 233 页)

定位精度较高，而且对不同的图像均具有较好的检测效果，实验结果充分验证了其有效性。

参 考 文 献

[1] 孙伟，夏良正，潘泓. 一种基于模糊划分的边缘检测算法[J]. 中国图象图形学报，2004，9(1)：18-22

[2] 章毓晋. 图像分析(第 2 版)[M]. 北京：清华大学出版社，2005

[3] 邵虹，张继武，崔文成，等. 基于图象内容的颅骨缺如自动分析研究[J]. 中国图象图形学报，2003，8(2)：214-218

[4] 张晶，张权，王欣. 一种新的基于统计向量和神经网络的边缘检测方法[J]. 计算机研究与发展，2006，43(5)：920-926

[5] 王侃伟，方宗德. 基于 Bubble 小波的多尺度边缘提取[J]. 计算机科学，2005，31(1)：272-274

[6] Yu Yuan-Hu, Chang Chin-Chen. A new edge detection approach based on image context analysis[J]. Image and Vision Computing, 2006, 24: 1090-1102

[7] Wu X, Memon N. Context-based, adaptive, lossless image coding [J]. IEEE Transactions on Communications, 1997, 45(4): 437-444