第 35 卷 第 8 期 计 算 机 工 程 2009 年 4 月
Vol.35 No.8 Computer Engineering April 2009
·软件技术与数据库· 文章编号：1000—3428(2009)08—0053—03 文献标识码：A 中图分类号：TP391

一种基于 FP_Tree 算法的决策树构造方法

徐林章，赵 强，张艳宁
(西北工业大学计算机学院，西安 710072)

摘 要：针对大规模训练元组决策树构造效率较低的问题，提出一种改进的决策树构造方法。该方法利用 FP_Tree 算法，比采用经典 Apriori 算法节省了更多内存开销。使用 FP_Tree 路径替代经典算法中训练元组的分裂计算，得到与原算法相同的决策树模型。实验结果证明，改进后的方法具有良好性能。

关键词：决策树；FP_Tree 算法；分类

Construction Method for Decision Tree Based on FP_Tree Algorithm

XU Lin-zhang, ZHAO Qiang, ZHANG Yan-ning
(College of Computer, Northwestern Polytechnical University, Xi'an 710072)

【Abstract】 Aiming at the low efficiency problem of the construction of decision tree in large-scale training units, this paper presents an improved construction method for decision tree. This method uses FP_Tree algorithm to save more memory than Apriori algorithm. It takes the place of split algorithm of the training units in classical algorithm by the path of FP_Tree, and gets the same decision tree model as the original algorithm. Test results show that the improved method has good property.

【Key words】 decision tree; FP_Tree algorithm; classification

1 概述

分类预测方法是数据挖掘技术中一项重要的分析方法。决策树[1-2]是一种常用、直观的分类算法，与其他分类算法相比，该算法构造出的决策树分类模型易于理解和解释，具有较高应用价值。但随着各类分析问题的日益复杂和数据规模的海量增长，传统决策树算法分类代价越来越高，且构造效率日趋降低，严重影响了后期分类预测效率。因此，如何提高决策树模型的构造效率成为当前分类方法研究中急需解决的问题。

本文提出一种将 FP_Tree[3-5]和决策树相结合的新方法，有效解决了对大规模训练元组数据构造决策树时效率偏低的问题，为决策树构造算法提供了改进思路。

2 决策树与 FP_Tree 算法

决策树归纳算法是根据类标记的训练元组数据集构造决策树，适合于探测式知识发现。其中，ID3 算法的应用较多，该算法从训练元组数据集以及与它们相关联的类标号开始，采用自顶向下递归的分治方法构造决策树。ID3 算法以信息增益作为属性选择度量。

FP_Tree 是文献[6]基于经典 Apriori[7]算法提出的改进算法，该算法将基于候选项集的频繁项挖掘问题转化成基于 FP_Tree 的挖掘问题，从而降低挖掘过程中频繁访问事务数据库的次数，很大程度上提高了频繁模式发现效率。构造 FP_Tree 只要访问 2 次事务数据库。第 1 次扫描数据库时，导出发生次数大于最小支持度的频繁项(1 项集)的集合和支持度计数(频率)，按出现次数对频繁 1 项集进行降序排列，以建立集合 L，并创建 FP_Tree 的根节点。第 2 次扫描事务数据库时，对每个事务的项按集合 L 的项次序进行处理，为每个事务在 FP_Tree 中建立一个分枝，不同事务可以共享前缀，每个节点由计数器记录对应项的发生次数。

本文用到的几个定义如下[6]：

定义 1 设数据划分 D 为带有类标记的元组的训练集。假定类标号属性具有 m 个不同值，定义 m 个不同的类 $C_i(i=1,2,\cdots,m)$。设 $C_{i,D}$ 是 D 中 C_i 类元组的集合，$|D|$和$|C_{i,D}|$分别是 D 和 $C_{i,D}$ 中元组的个数。

设节点 Q 代表或存放划分 D 的元组。选择具有最高信息增益的属性作为节点 Q 的分裂属性。对 D 中元组分类所需的期望信息由下式给出：

$$Info(D)=-\sum_{i=1}^{m}P_i\mathrm{lb}P_i \tag{1}$$

其中，P_i 是 D 中任意元组属于类 C_i 的概率，并用$|C_{i,D}|/|D|$估计；$Info(D)$ 是识别 D 中元组的类标号的平均信息量，又称为 D 的熵。

定义 2 假设按属性 A 划分 D 中的元组，根据训练数据的观测，属性 A 具有 v 个不同的值，即 $\{a_1,a_2,\cdots,a_v\}$。如果 A 是离散值的，则其值直接对应在 A 上测试的 v 个输出，可以用属性 A 将 D 划分成 v 个子集，即$\{D_1,D_2,\cdots,D_v\}$，其中，D_j 包含 D 中的元组，它们在 A 上具有值 a_j。上述划分对应从节点 Q 生长出来的分枝。经过划分后，为了得到准确分类，根据下式度量需要的信息量：

$$Info_A(D)=-\sum_{j=1}^{v}\left|D_j\right|/\left|D\right|\times\mathrm{lb}P_i \tag{2}$$

其中，项$|D_j|/|D|$是第 j 个划分的权重；$Info_A(D)$ 是基于按 A 划分、对 D 的元组进行分类时需要的期望信息量，其值越小，

作者简介：徐林章(1966－)，男，博士研究生，主研方向：数据库，数据挖掘；赵 强，硕士研究生；张艳宁，教授

收稿日期：2008-11-29 **E-mail：**xlz0088@sina.com

划分纯度越高。

定义 3 信息增益定义为原来的信息需求(即仅基于类比例)与新需求(对 A 划分后得到)之间的差，即

$$Gain(A) = Info(D) - Info_A(D) \tag{3}$$

选择具有最高信息增益的属性 A 作为节点 Q 的分裂属性，即按能做“最佳分类”的属性 A 进行划分，使完成元组分类所需要的信息量最小，即最小化 $Info_A(D)$ 。

3 基于 FP_Tree 算法的决策树构造方法

3.1 构造原理

基于 FP_Tree 的决策树构造方法先对训练元组数据集构造 FP_Tree，再对 FP_Tree 从叶节点到根节点的每条路径进行分解，从而得到各属性值和类别标号的一个组合。FP_Tree 叶节点的计数为相应路径的发生次数，根据各路径发生次数来计算信息增益，可以有效避免原始训练元组在递归过程中的传递，节约内存资源。

3.2 主要步骤

基于 FP_Tree 算法的决策树构造方法流程如图 1 所示。

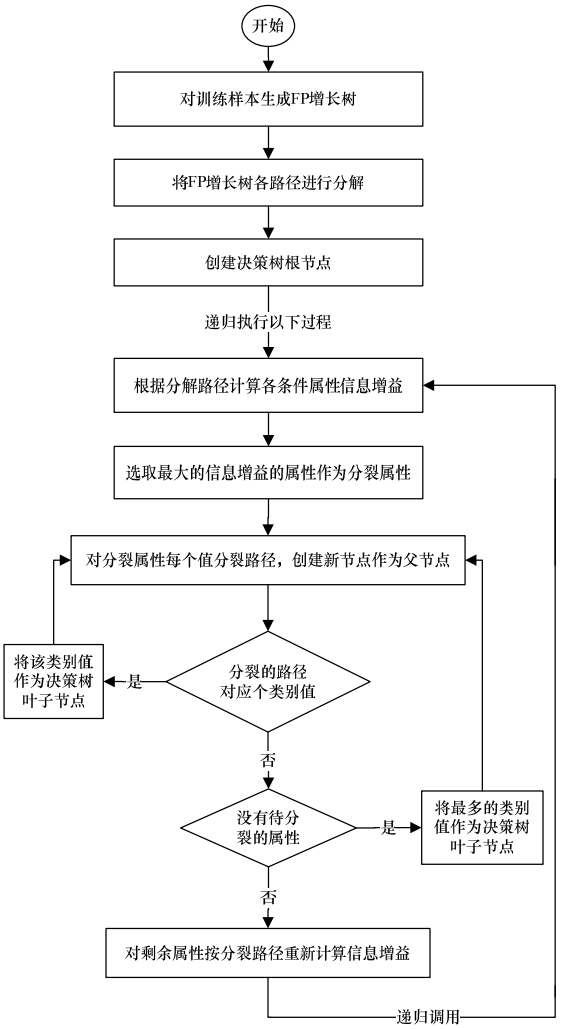

图 1 基于 FP_Tree 算法的决策树构造方法流程

3.3 算法分析

基于 FP_Tree 算法的决策树构造方法产生的决策树模型与经典 ID3 算法的结果一致，且具有以下优点：

(1)ID3 算法用训练元组数据计算信息增益，设训练元组数量为 N，条件属性数量为 V，算法效率与训练元组的数据量大小和属性数量正相关，则算法的计算量为 $G_1 = N \times V$ 。

本文方法采用频繁发生项集(即 FP_Tree 路径)代替训练元组计算信息增益量，由于执行效率主要与 FP_Tree 路径数相关，因此得到极大提高，具体分析如下：

1)假设训练元组数量为 N，FP_Tree 路径数量为 M，有 $M \leqslant N$ 。

2)设训练元组有 V 个属性，每个属性具有$\{d_1,d_2,\cdots,d_v\}$个不同值，则 $M \leqslant \prod_{i=1}^{v} d_i$ 。

本文方法的计算量根据下式计算：

$$G_2 = N \times 2 + M \times V \tag{4}$$

2 种算法的计算量比较如下式所示：

$$G_2 / G_1 = (N \times 2 + M \times V) / N \times V = 2/V + M/N \tag{5}$$

由式(5)可以看出，当训练元组规模较大时，数据有一定规律，路径数量相比训练元组数量将极大减少，即 $M<<N$。此时，$G_2/G_1<<1$，即本文方法的计算量远小于 ID3 算法。

(2)在 FP_Tree 构造过程中，第 1 步生成的频繁项目集(1 项集)必须满足最小支持度，因此，设置合理的最小支持度参数能有效规避数据中的噪声和离群点，排除训练元组中的异常情况，起到先剪枝的作用，从而降低决策树规模和复杂度。

4 实验结果与分析

4.1 计算机仿真计算

通过引入经典的气候、温度等属性，构造是否比赛决策树的数据，并验证本文方法和 ID3 算法的结论是否一致，具体步骤如下：

(1)根据如图 2 所示的比赛数据构造 FP_Tree，FP_Tree 路径结果如图 3 所示。

outlook	temperature	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rain	mild	high	false	yes
rain	cool	high	false	yes
rain	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rain	mild	high	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rain	mild	high	true	no

(a)原始数据

1	2	3	4	5
1	1	1	1	2
1	1	1	2	2
2	1	1	1	1
3	2	1	1	1
3	3	1	1	1
3	3	2	2	2
2	3	2	2	1
1	2	1	1	2
1	3	2	1	1
3	2	1	1	1
1	2	2	2	1
2	2	1	2	1
2	1	2	1	1
3	2	1	2	2

(b)转换后的数据

图 2 比赛数据

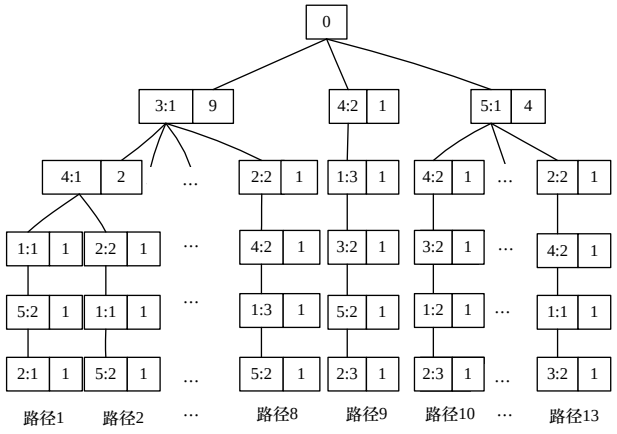

图 3 FP_Tree 路径结果

(2)对 FP_Tree 的路径进行分解，图 4 描述了分裂属性的选择步骤。

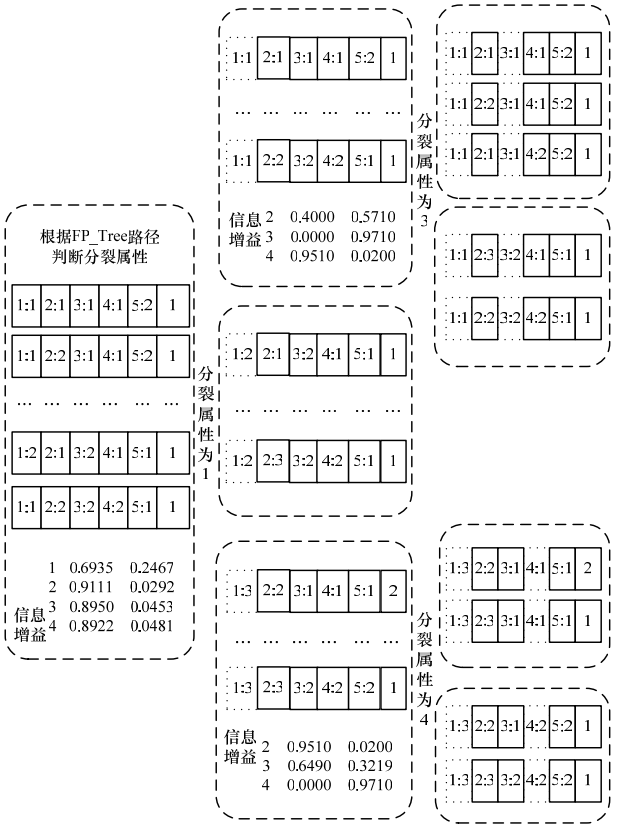

图 4　分裂属性的选择步骤

(3)最终构造的决策树如图 5 所示。

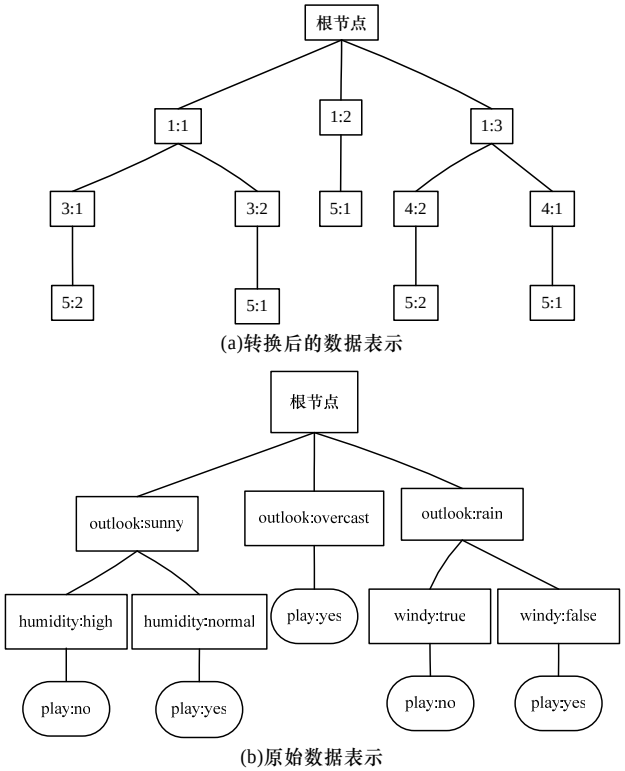

图 5　决策树

本文方法最终构造的决策树与 ID3 算法的结果一致。

4.2　较大规模税务数据算法实验

较大规模税务数据算法实验针对某地税局的实际纳税人数据，根据纳税人的纳税总额、主营业务收入等属性构建纳税人是否欠税的决策树模型。用本文方法设置不同最小支持度参数，在相同运行环境下，与 ID3 算法执行时间进行比较，以确定本文方法是否能提高效率。实验使用的训练样本包括 2 组：(1)3 个分局纳税人数据 17 157 条；(2)全市地税局纳税人数据 51 472 条。

实验结果如表 1 和表 2 所示。

表 1　第(1)组训练样本的实验结果

算法	路径数	执行时间/s
经典算法	-	182.250
本文方法(3%支持度)	3 707	63.610
本文方法(5%支持度)	3 602	57.078
本文方法(7%支持度)	2 948	31.265

表 2　第(2)组训练样本的实验结果

算法	路径数	执行时间/s
经典算法	-	749.578
本文方法(3%支持度)	4 477	75.844
本文方法(5%支持度)	4 342	75.672
本文方法(7%支持度)	3 400	49.547

由实验结果可以看出，本文方法能显著减少内存消耗和 CPU 执行时间，从而有效提高决策树构造效率。随着训练元组数据集规模的增大，效率的提高效果越来越显著。

5　结束语

本文采用 FP_Tree 路径代替训练元组来计算属性的信息增益，从而选择构造决策树的分裂属性。改进后的方法可以有效减少内存开销、提高决策树模型构建效率。

参考文献

[1] 鲁　为，王　枞. 决策树算法的优化与比较[J]. 计算机工程，2007, 33(16): 189-190.

[2] 高　静，徐章艳，宋　威，等. 一种新的基于粗糙集模型的决策树算法[J]. 计算机工程, 2008, 34(3): 9-11.

[3] Han Jiawei, Pei Jian, Yin Yiwen. Mining Frequent Patterns Without Candidate Generation[C]//Proc. of ACM-SIGMOD Int'l Conf. on Management of Data. Dallas, USA: ACM Press, 2000.

[4] 宋余庆，朱玉全，孙志挥，等. 基于 FP_Tree 的最大频繁项目集挖掘及更新算法[J]. 软件学报, 2003, 14(9): 1586-1592.

[5] 郭宇红，童云海，唐世渭，等. 基于 FP_Tree 的反向频繁项集挖掘[J]. 软件学报, 2008, 19(2): 338-350.

[6] Han Jiawei, Kanmber M. Data Mining: Concepts and Techniques[M]. Morgan, USA: Morgan Kaufmann, 2005.

[7] Borgelt C. Efficient Implementations of Apriori and Eclat[C]//Proc. of Workshop on Frequent Item Set Mining Implementations. Melbourne, USA: [s. n.], 2003.

编辑　陈　晖