

两种决策树的事前修剪算法

屈俊峰, 朱 莉, 胡 斌

(中国地质大学 计算机学院, 湖北 武汉 430074)

(Qmxwt@163.com)

摘 要:修剪决策树可以在决策树生成时或生成后,前者称为事前修剪。决策树上的每一个节点对应着一个样例集,通过分析样例集中样例的个数或者样例集的纯度,提出了基于节点支持度的事前修剪算法 PDTBS 和基于节点纯度的事前修剪算法 PDTBP。为了达到修剪的目的,PDTBS 阻止小样例集节点的扩展,PDTBP 阻止高纯度样例集节点的扩展。分析表明这两个算法的时间复杂度均呈线性,最后使用 UCI 的数据实验表明:算法 PDTBS, PDTBP 可以在保证分类精度损失极小的条件下大幅度地修剪决策树。

关键词:决策树;事前修剪;支持度;纯度

中图分类号: TP18 **文献标识码:** A

Two algorithms of pre-pruning decision tree

QU Jun-feng, ZHU Li, HU Bin

(Institute of Computer Science, China University of Geosciences, Wuhan Hubei 430074, China)

Abstract: Pruning decision tree may occur in the process of creating decision tree or after that, the former is called pre-pruning. Every node on decision tree has a corresponding sample set. By analyzing the quantity of sample in the sample set or the purity of it, algorithm PDTBS, viz. pre-pruning decision tree based on support, and algorithm PDTBP, viz. pre-pruning decision tree based on purity were put forward. For pre-pruning, PDTBS prevented the node of a small sample set from extending; PDTBP prevented the node of a high purity sample set from extending. The time complexities of two algorithms were analyzed linear. Experiment results on UCI data show that the two algorithms can pre-prune decision tree to a great extent, while all its accuracy hardly diminishes.

Key words: decision tree; pre-pruning; support; purity

决策树为机器学习中的分类问题提供了一种解决方法。通过学习训练样例集合归纳得出的决策树能够分类后续类别未知的例子。在生成决策树的经典算法 ID3^[1]被提出后,研究工作主要集中在:非叶节点相关属性的选择标准、分类精度的提高、修剪决策树等方面。文献[2]给出了修剪决策树的理论依据:树的复杂性和树的分类精度不完全成正比,过于复杂的树甚至有可能降低分类精度。决策树修剪^[3,4]分为事前修剪(pre-pruning)和事后修剪(post-pruning)。事后修剪^[5]以提高树的分类精度为目的在决策树生成之后进行,花费了额外的时间;事前修剪在生成决策树的同时进行,在对分类精度影响不大的条件下,简化了决策树,同时由于没有额外的步骤,提高了效率。文献[1]利用卡方测试(chi-square test)做事前修剪,本文则提出了另外两种事前修剪的算法。

1 事前修剪

决策树的非叶节点表示属性;节点的向下分支表示对应的属性值;叶节点表示类别。节点对应的样例集是指通过从整棵树的根节点开始到此节点为止的所有“属性-值”测试的训练样例集合。决策树的生成通常从根节点开始,选择节点相关属性,然后根据属性值的个数向下伸出相应数量的分支,形成子节点,如此循环进行下去,直到满足下面三个条件时,节点停止扩展,成为叶节点:1)节点对应样例集中所有样例

均为一类,那么以此类标记此节点;2)节点对应样例集为空集,那么以其父节点对应样例集中最普遍的样例类别标记此节点;3)从根节点到此节点的父节点所有的属性均被用过一次,那么处理方法同第二种情况。通过这种方法生成的决策树是完整的,已经完全展开,没有进一步生长的条件了。

事前修剪是通过抑制生成一棵完整的树来进行简化。普遍的做法是设置一个阈值,然后在扩展每个节点之前,考察树的某一指标或者已被确定的节点相关属性的某一指标,如果指标达不到阈值,则此节点停止向下扩展、作为叶子并用此处最普遍的样例类别标记。

最简单的方法是设定一个深度阈值,当树的节点达到这个深度时就停止向下扩展。文献[1]采用卡方测试来判断节点相关属性的分类能力是否能达到最低要求。假设一分类问题仅有两个类别:正例和反例。设某节点对应的样例集 C 中有 p 个正例和 n 个反例,且确定了 A 作为此节点的相关属性。再设 A 有 k 个属性值,则按照“属性-值”测试样例集 C ,它们被分为 k 个子集 $\{C_1, C_2, C_3, \dots, C_k\}$, C_i 包含 p_i 个正例和 n_i 个反例。如果属性 A 的分类能力不强,那么其中一个表现是 C_i 的正反例所占比例与 C 的差异不大,即 p_i 的估计值 p_i' 为:

$$p_i' = p_i \times ((p_i + n_i) / (p + n)) \quad (1)$$

n_i 的估计值 n_i' 为:

$$n_i' = n_i \times ((p_i + n_i) / (p + n)) \quad (2)$$

收稿日期:2005-09-08 修订日期:2005-12-09

作者简介:屈俊峰(1980-),男,湖北襄樊人,硕士研究生,主要研究方向:人工智能;朱莉(1952-),女,江西永新人,教授,主要研究方向:人工智能、网络数据库;胡斌(1978-),男,湖北武汉人,硕士研究生,主要研究方向:机器学习。

构造统计量:

$$chi = \sum_{i=1}^k \left((p_i - p_i')^2 / p_i' + (n_i - n_i')^2 / n_i' \right),$$

$$chi \in [0, p + n] \quad (3)$$

按照统计学的分析^[6], chi 近似服从自由度为1的卡方分布(如图1)。此处可以简单地认为 chi 即表示属性 A 的分类能力, chi 越大属性 A 的分类能力就越强。所以可以设置一个阈值, 当属性 A 的分类能力达不到此值时节点就停止向下扩展。设所有属性的平均属性值个数为 k , 那么用卡方测试判断一个节点是否需要扩展的时间复杂度为 $O(k)$ 。

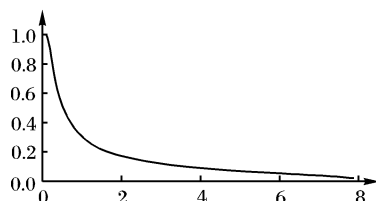


图1 自由度为1的卡方分布

2 基于节点支持度的事前修剪算法

决策树上节点的支持度 (support) 表示节点对应的样例集中训练样例的个数。决策树上各节点的支持度是不一样的, 在根节点处由于没有经过任何“属性-值”测试, 它包括所有的训练样例, 其支持度最高; 子节点由于比父节点多进行了一次“属性-值”测试, 它所对应的样例集是父节点对应样例集的子集, 故子节点的支持度小于等于父节点的。表1表示训练样例集, 图2是由ID3生成的决策树, 每个节点左边的值表示其支持度。节点的支持度越大, 则表明利用从根节点到此节点这一路径进行分类的样例越多, 据此可认为以后的分类任务中此路径的使用较多, 价值较大。利用这一特性, 在决策树的生成过程中, 可以阻止支持度小的节点向下扩展。即可定义一支持度阈值, 支持度超过此值的节点可以向下扩展; 否则就停止向下扩展, 将其作为叶节点并以此节点处最普遍的样例类别标记此节点。由此整理得出基于节点支持度的事前修剪算法 (Pre-pruning Decision Tree Based on Support, PDTBS):

- 1) 设置支持度阈值 $support$, 根节点 $root$ 进入队列 Q ;
- 2) 如果 Q 为空, 那么返回 $root$; 否则取出 Q 首元素: 节点 N ;
- 3) 确定 N 对应的样例集 S , 根据 S 计算出 N 的支持度 sup ;
- 4) 如果 N 满足停止扩展的三个条件, 则将其作为叶节点并做相应的处理, 转2);
- 5) 如果 sup 小于 $support$, 则 N 不扩展, 作为叶节点并以 S 中最普遍的样例类别标记它; 否则按照某一标准 (如信息增益最大), 确定 N 的相关属性 A , 根据 A 的属性值个数向下伸出相应数量的分支, 形成子节点, 这些子节点依次进入 Q ;
- 6) 转2)。

算法 PDTBS 中, 先进先出队列 Q 用于存放待考察是否需要扩展的节点。图3表示将支持度阈值设置成3, 节点相关属性的选择以信息增益最大为标准, 算法 PDTBS 生成的决策树。

一般可以把支持度阈值设置成一个绝对数值, 但是这样做的局限是此数值要和训练样例集的大小 (样例个数) 密切相关, 需不断调整。较好的做法是将其设置成整个训练集样例个数 (设为 $|S|$) 的一个百分比 (如 $|S| \times 5\%$)。

表1 训练样例集, 4个属性 (A), 2种类别 (C)

A1	A2	A3	A4	C	A1	A2	A3	A4	C
O	H	H	F	P	O	M	H	T	P
R	M	H	F	P	O	H	N	F	P
R	C	N	F	P	S	H	H	F	P
O	C	N	T	P	S	H	H	T	N
S	C	N	F	P	R	C	N	T	N
R	M	N	F	P	S	M	H	F	N
S	M	N	T	P	R	M	H	T	N

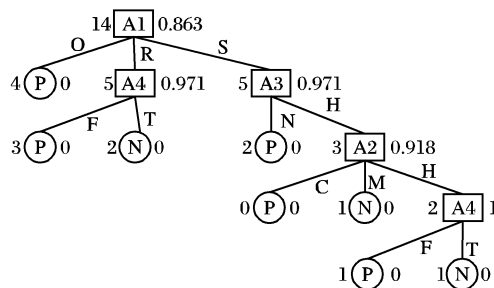


图2 表1对应的决策树(由ID3生成)

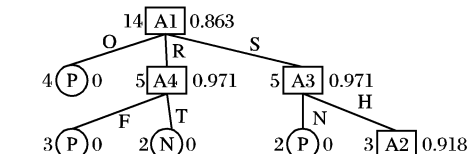


图3 表1对应的决策树(由PDTBS生成, $support = 3$)

3 基于节点纯度的事前修剪算法

决策树上节点的纯度 (purity) 表示节点对应的样例集中各种类别样例所占的比例情况。如果样例集中的所有样例均为同类, 那么这个集合就是完全纯洁的, 其纯度最高。纯度对于只有两种类别的样例集可以用正反例的比例简单地表示。如果样例的类别数大于等于3, 则可以引入信息论中的熵, 定义如下:

$$Entropy(S) = \sum_{i=1}^c -P_i \log_2 P_i, Entropy(S) \in [0, \log_2 C] \quad (4)$$

其中 S 表示待求熵的样例集, C 表示类别个数, P_i 表示各个类别样例在 S 中所占比例。熵越小则样例集就越纯。图2中, 每个节点右边的值表示纯度 (熵)。如果节点对应样例集的纯度足够高并达到设定的阈值 (即其熵小于此值), 那么就可以考虑停止此节点向下扩展。因为: 1) 较高纯度节点继续扩展容易造成决策树对训练样例的过度拟合^[7], 从而导致对真实规律归纳的偏离; 2) 阻止这种扩展可以有效避免由噪声导致的复杂的树的生成。

只需将算法 PDTBS 中的 1) 改为“设置纯度阈值 $purity$ ……”, 3) 改为“……根据 S 计算出 N 的纯度 (即熵) pur ”, 5) 改为“如果 pur 小于 $purity$ ……”, 就形成了基于节点纯度的事前修剪算法 (Pre-pruning Decision Tree Based on Purity, PDTBP), 其中纯度阈值通常设置成最大熵值的一个百分比 (如 $\log_2 C \times 60\%$)。

需指出的是: 1) PDTBP 是一种贪婪算法, 可能会陷入局部最优; 2) PDTBP 算法并不适用于小训练样例集生成决策树。如图2中节点右边的数值表示熵, 可以发现非叶节点的熵都很大 (即样例集的纯度都不是足够的高), 即使取纯度阈值为 $\log_2 2 \times 60\% = 0.6$ 也没有任何分支被修剪。

4 算法分析与对比实验

决策树的生成和事前修剪是两个交织在一起的过程,当树生成结束时,事前修剪也就结束了。基于节点支持度、纯度的事前修剪算法 PDTBS、PDTBP 简单易行,它们仅仅需要在决策树的节点上增加一(数)个数据域,用于记录节点对应样例集中样例的总(各个类别的)数目。前者在判断节点是否需要扩展时只需一次比较,其时间复杂度为 $O(1)$;后者需在比较前计算一下熵的值,其时间复杂度为 $O(C)$, C 为类别个数。由于引入了队列,使得算法 PDTBS、PDTBP 消除了传统决策树生成算法(如 ID3)中的递归调用,提高了效率;同时,这两个算法并没有对节点相关属性的选择标准做硬性的规定,在实际使用时可以根据需要定义,比较灵活。

采用 UCI^[8] 中关于机器学习的 tic-tac-toe 数据库做对比实验。此库中的记录分为两个类别共 958 条,每次实验随机抽取 70% 的记录作为训练样例,分别用 ID3、卡方测试(阈值分别设为 1, 2, 4)做事前修剪的 ID3、PDTBS 算法(支持度阈值分别设为 958 的 0.5%, 1%, 1.5%; 节点相关属性的选择以信息增益最大为标准)、PDTBP 算法(纯度阈值分别设为 0.2, 0.4, 0.6; 节点相关属性的选择以信息增益最大为标准)生成 10 棵决策树,对剩下 30% 的记录分类。独立地实验 100 次,统计得出这 10 棵决策树的平均节点数与平均分类精度于表 2 中,在此表中还列出了判断一个节点是否需扩展的时间复杂度。

表 2 分类实验结果

算法	阈值	平均分类精度(%)	平均节点数	时间复杂度
ID3		84.681	275	$O(1)$
ID3 (Chi-square test)	1	84.764	268	$O(K)$
	2	84.685	253	
	4	82.991	158	
PDTBS	0.5%	84.244	197	$O(1)$
	1%	82.291	135	
	1.5%	80.092	105	
PDTBP	0.2	84.681	275	$O(C)$
	0.4	84.579	269	
	0.6	82.881	208	

其中, K 为属性的平均属性值个数; C 为类别个数。

从表中可以发现,卡方测试的方法对树的修剪较为保守,但精度控制得好;PDTBS 算法对树的修剪幅度最大,但相对

于树的大幅修剪(例如 $support = 1.5\%$, 相对于 ID3 生成的树,节点从 275 变为 105,树被修剪了 61.8%),精度并没有如此幅度的降低(上例,树的精度从 84.681% 变为 80.092%,仅仅降低了 4.589%),并且此方法的时间复杂度最小;使用 PDTBP 算法时要注意对阈值的选择,例如表中当 $purity = 0.2$ 时树根本没有被修剪,这种方法对树的修剪最保守,但精度控制得较好。

5 结语

假设训练样例有 n 个属性,每个属性平均有 m 个属性值,那么生成的一棵决策树,理论上其节点数可达 m^n 个,这是一个很巨大的数字。事前修剪的目的就在于控制决策树的规模,避免生成较大的树,这样做一方面节省了空间,另一方面在利用它分类时也节省了时间。虽然过于复杂的树降低分类精度的可能性存在^[2]并在实验中再次得到了验证(表 2 中卡方测试的第 1 行),但绝大多数事前修剪的方法都会降低分类精度,关键是要在树的复杂性和分类精度之间取得一种平衡,即要求在(大幅度)修剪决策树的同时精度损失要尽可能的小,本文提出的事前修剪算法 PDTBS 和 PDTBP 都很好地达到了这一目的。

参考文献:

- [1] QUINLAN JR. Induction of Decision Tree[J]. Machine Learning, 1986, 1(1): 81-106.
- [2] FAYYARD UM, IRAN KB. What Should Be Minimized in a Decision Tree?[A]. Proceedings of Eighth National Conference on Artificial Intelligence[C]. Boston. MA. USA, 1990. 749-754.
- [3] QUINLAN JR. Simplifying decision trees[J]. International Journal of Man-Machine Studies, 1987, 27(3): 221-234.
- [4] BRESLOW LA, AHA DW. Simplifying decision trees: a survey[J]. Knowledge Engineering Review, 1997, 12(1): 1-40.
- [5] WEI HL. Comparison among Methods of Decision Tree Pruning[J]. Journal of Southwest Jiaotong University, 2005, 40(1): 44-48.
- [6] HOGG RV, CRAIG AT. Introduction to mathematical statistics [M]. London: Collier-Macmillan, 1970.
- [7] MITCHELL TM. Machine Learning [M]. China Machine Press, 2003.
- [8] University of California. Irvine repository of machine learning database[DB/OL]. <ftp://ics.uci.edu/inthe/pub/machine-learning-databases/directory>, 2005-05-12.

(上接第 637 页)

策表的有效算法,并已在实际应用中得到了广泛运用。而基于模糊粗糙集紧计算域的属性约简算法 CCD-FRSAR 是最新提出的,本文系统地研究并比较了这两种算法的特点及性能。我们的理论分析和实验计算结果均表明:CCD-FRSAR 在时间复杂度和计算结果的可靠性上均优于 FRSAR。在算法的收敛性上,算法 FRSAR 无法保证总可以在有限时间内收敛,而算法 CCD-FRSAR 利用剪枝的思想方法,使得各搜索层剩余条件属性集合的势总是向减小的趋势发展,从而保证了收敛。

需要指出的是,尽管 CCD-FRSAR 有这些优点,但它并没有考虑到大型数据集分析中,由于人为错误或噪声可能导致的某些数据被误分的分类问题,缺乏抗噪声干扰的能力,这将在一定程度上制约其处理复杂应用问题的有效性。

参考文献:

- [1] SHEN Q, CHOUCOULAS A. A modular approach to generating fuzzy rules with reduced attributes for the monitoring of complex systems[J]. Engineering Applications of Artificial Intelligence, 2000, 13(3): 263-278.
- [2] JENSEN R, SHEN Q. Fuzzy-rough attribute reduction with application to web categorization[J]. Fuzzy Sets and Systems, 2004, 141(3): 469-485.
- [3] JENSEN R, SHEN Q. Fuzzy-rough data reduction with ant colony optimization[J]. Fuzzy Sets and Systems, 2005, 149(1): 5-20.
- [4] BHATT RB, GOPAL M. On fuzzy-rough sets approach to feature selection[J]. Pattern Recognition Letters, 2005, 26(7): 965-975.
- [5] RADZIKOWSKA AM, KERRE EE. A comparative study on fuzzy-rough sets [J]. Fuzzy Sets and Systems, 2002, 126(2): 137-155.