文章编号:1001 - 9081(2003)11 - 00 - 0

基于决策树的不完全决策表的数据补充及规则提取

文硕频,乔胜勇,陈彩云,李治国
(南开大学 组合数学研究中心,天津 300071)

摘 要:不完全信息系统中遗失数据的补充和规则的提取,一直是数据挖掘技术面临的重要问题。文中给出了一种基于决策树来求解此问题的算法。对于给定的不完全决策表,该算法应用改进的 ID3 算法来构造决策树,在构造决策树的过程中对遗失值进行补充。对于不能在决策树上补充的遗失值,定义了一种相关对象之间的相似度来填充。该算法简单,易于操作。

关键词:不完全决策表;遗失值;数据补充;决策树;规则提取
中图分类号: TP182 **文献标识码**:A

Decision-Tree-based Missing Data Filling and Rules Extraction in Incomplete Decision Table

WEN Shuo-pin, QIAO Sheng-yong, CHEN Cai-yun, LI Zhi-guo
(Center for Combinatorics, Nankai University, Tianjin 300071, China)

Abstract: Missing data filling and rules extraction in incomplete decision table are two important data mining problems. Based on decision tree, the paper gives an algorithm to solve these problems. For a given incomplete decision table, the algorithm constructs decision tree using the improved ID3 algorithm, and fills the missing data in the process of constructing the decision tree. A similar measure to fill the missing data that can't be filled in the process of constructing the decision tree is defined. The algorithm is simple and easily handled. The algorithm is illuminated with an example.

Key words: incomplete decision table; missing data; data filling; decision tree; rule extraction

在现实生活中,由于历史和人为的原因,数据中存在遗失值的现象是不可避免得。在很多情况之下,信息系统都是不完全的。对不完全信息系统的遗失数据的处理问题是一个古老的分析任务,如果将具有遗失值的数据从信息系统中删除,不仅会造成资源的大量浪费,更会丢失隐藏在数据中的数据挖掘任务所寻求的知识点,但是,对遗失值不正确的填充又往往将新的噪声引入数据中,使挖掘任务产生错误的结果。因此,如何正确的对遗失数据进行填充是预处理过程中的一个重要问题。

Tzung-Pei Hong 等给出了一种基于粗糙集的上近似和下近似在对不完全决策表进行规则提取的同时补充遗失值的算法[1]。该算法计算量大且繁琐,并且也不是所有的遗失值都可以补充。对于不完全决策表,本文通过构造决策树来实现规则提取;在构造决策树的过程中,寻找数据的决策属性之间的关系,对遗失值进行补充;对于不能在决策树上补充的遗失值,定义了一种相关对象之间的相似度来填充。

1 相关知识简介

信息系统由二元集组成,记为 $S = < U, A >$,其中 U 是由 n 个研究对象 $\{x_1, x_2, \ldots, x_n\}$ 组成的非空集合,U 中的每个元素称为对象;A 是由 k 个属性 $\{a_1, a_2, \ldots, a_k\}$ 组成的有限非空集合,并且对 $a \in A$,存在关系 $a: U \to V_a$,V_a 称为属性 a 的值域。如果至少有个一个属性 $a \in A$,使得 V_a 含有空值(遗失值),则称 S 为一个不完全信息系统,否则它是完全的,我们用“*”表示空值。不完全决策表是一类特殊而重要的不完全信息系统。不完全决策表是一个具有形式 $S = < U, A \cup \{d\} >$ 的不完全信息系统,其中 $d \notin A$ 且 $* \notin V_d$ 称为决策属性,A 中的元素称为条件属性。假设 d 的值域 V_d 的基数为 r,则决策属性 d 决定了 U 的一个划分 $U = C_1 \cup \ldots \cup C_r$,其中 $\forall x, y \in C_i, d(x) = d(y)$,对所有的 $1 \le i \le r$。C_i 称为 S 的第 i 个决策类。

对于每一个条件属性 a,我们定义对象之间的不完全相似关系如下:

$$\mathrm{SIM}(a) = \{(x,y) \in U \times U \mid a(x) = a(y) \text{ 或 } a(x) = * \text{ 或 } a(y) = *\}$$

令 $S_a(x) = \{y \in U \mid (x,y) \in \mathrm{SIM}(a)\}$,即 $S_a(x)$ 是与 x 可能不可区分的对象的最大集合。则 $U/SIM(a) = \{S_a(x) \mid x \in U\}$ 表示了 U 的一种分类,它们一般不构成 U 的划分,但它们构成 U 的一个覆盖。

2 决策树构造

决策树方法自 20 世纪 60 年代以来,在机器学习、数据挖

收稿日期:2003 - 05 - 27;**修订日期**:2003 - 09 - 01 **基金项目**:国家 973 计划资助项目(G19980306)

作者简介:文硕频(1979 -),女,湖南桃江人,硕士研究生,主要研究方向:组合数学、计算机软件; 乔胜勇(1978 -),男,河北衡水人,硕士研究生,主要研究方向:组合数学、计算机软件; 陈彩云(1975 -),女,江苏南通人,博士研究生,主要研究方向:组合数学、数据挖掘; 李治国(1977 -),男,河北张家口人,硕士研究生,主要研究方向:组合数学、计算机软件.

掘等领域得到了广泛的应用。对一个分类问题或规则学习问题,决策树是一树状结构。它的生成是一个从上至下、分而治之的过程。它从根结点开始,对决策表的一个条件属性进行测试,根据不同的结果将对象集划分成不同的对象子集,每个对象子集构成一个子节点,对每个子节点再进行划分,生成新的子节点,不断循环,直到达到特定的终止准则[5]。生成的决策树每个叶子节点对应一个分类。很容易从决策树中提取规则,从根到叶子,自上而下的每条数据创建一个规则。

设对象集为 U,决策树的生成算法可以描述如下:

(1) 创建一个根结点 N;

(2) 如果对象集 U 的每一个对象都同属于一个决策类,则该结点成为叶子节点,并用该类标记;否则,选取某个条件属性 a,将 U 划分成 n 个子集,对每个子集重复(2) 的判断。

(3) 当不再有条件属性可以选择或所有节点都成为了叶子节点时,生成了最后的决策树。

上述算法中选择进行测试的条件属性是不确定的,对于不同的选择,可能导致不同的决策树,从而得到不同的决策规则。一般认为,决策树的结构越简单[3],可以认为在更本质的层次上概括了事务的规律。从最短描述长度的准则考虑,生成最少叶子且每个叶子深度最小的决策树是最优决策树,这也是许多算法研究的热点。最优决策树的确定已被证明是 NP-Hard 问题。

因此,在每一个节点上如何选择进行测试的条件属性是影响最后生成的决策树结构的关键问题。在构造决策树的测试评价标准中最常用的是 ID3 算法中所采用的信息增益(information gain)方法[2]。我们将这种方法进行推广,来实现对不完全决策表构造优化的决策树。

根据信息理论,应用到不完全决策表上有:

$$I(U) = -\sum_{i=1}^{r}\frac{p(C_i,U)}{|U|}\log_2\frac{p(C_i,U)}{U} \tag{1}$$

$$E(a) = \sum_{i=1}^{m}\frac{|U_i|}{|U|}I(U_i) \tag{2}$$

其中 $I(U)$ 表示一棵决策树能对样本作出(不完全) 正确划分的期望熵,$E(a)$ 表示以 a 为节点所需的期望信息,$p(C_i, U)$ 表示 U 中对象属于第 C_i 类的概率,$|\ |$ 为集合的元素的个数(基),对数函数以 2 为底,因为信息用二进制编码。则备选属性 a 的信息增益为:

$$G(a) = I(U) - E(a) \tag{3}$$

设

$$IV(a) = -\sum_{i=1}^{m}\frac{|U_i|}{|U|}\log_2\frac{|U_i|}{U} \tag{4}$$

$IV(a)$ 是关于属性 a 取值的信息度量,这样定义属性 a 的信息增益率为

$$GR(a) = \frac{G(a)}{IV(a)} \tag{5}$$

ID3 算法构造决策树时采用信息增益作为选择属性的测试标准(选取信息增益最高的属性作为内部节点将对象子集进行下一步的分类),Quinlan 通过试验[3],对若干属性测试标准做比较,发现信息增益率能得到更好的决策树,而且不易产生不平衡的划分。因此,我们在对不完全决策表构造决策树的算法中将采用信息增益率作为每个节点选择测试属性的判断标准,并且 $|U_i|$ 只计算 U_i 中取值确定的对象的个数。

3 遗失值的补充

不完全信息系统的遗失值的处理是数据挖掘预处理过程的一个重要研究问题。对于遗失值的处理有很多方法[2]:最常用的方法是使用最可能的值填充空缺值。与其他方法相比,它是用现存数据的多数信息来推测遗失值。本文中,我们利用上述方法的思想,在构造决策树的过程中,通过我们定义的规则对遗失值补充,当决策树构建完毕,遗失值也填充完毕,再从决策树上提取(不完全) 决策表的规则。

因为不完全决策表中存在不确定的信息,对于属性 a,为了区分某个对象在此属性的取值是不是确定的,在构造决策树的过程中,当采用属性 a 作为测试属性来对对象集 U 进行分类时,在子节点中含的对象中,我们用 x^* 表示对象 x 在条件属性 a 的取值遗失,需要进行补充。对于由属性 a 划分得到的子节点中含 $*$ 对象 x_i^* 的取值的确定,我们依次采用如下两种方法:

(1) 如果存在某个子节点包含的所有对象都属于同一个决策类,则首先将该节点中所有 x_i^* 在 a 的取值确定为该节点其他不含 $*$ 对象在属性 a 的取值(显然,所有不含 $*$ 对象在 a 的取值都一样),再将 x_i^* 修改为 x_i,并且从其他节点中去掉 x_i^*,否则

(2) 对于 x_i^* 我们定义 U 的一个子集合 B:

$$B = \{x \in U \mid a(x) \neq *\}$$

即 B 为在属性 a 的取值确定的所有对象的集合,然后我们定义对象 x_i^* 与 B 中对象 x_j 的"相似度" 如下:

$$S(x_i,x_j) = \frac{|A_{ij}|}{|A|} + \delta_{ij}$$

其中 $A_{ij} = \{a \in A \mid a(x_i) = a(x_j)\}$ 为 x_i 和 x_j 取值相同且确定的属性的集合,A 为所有属性的结合,$|\ |$ 为集合中的元素的个数(基),

$$\delta_{ij} = \begin{cases}1, & d(x_i) = d(x_j)\\ 0, & d(x_i) \neq d(x_j)\end{cases}$$

取出与 x_i 具有最大相似度的 B 中的对象 x_j,将 x_j 的属性 a 的取值作为 x_i 的取值,在含 x_j 的子节点中修改 x_i^* 为 x_i,并且从其它节点中去掉 x_i^*。如果节点中还有带 $*$ 的对象,则重复 2 的过程,直到所有子节点中对象都不含有 $*$。

此时,U 中对于属性 a 具有不确定值的对象的属性值都被确定了。停止对属性值的修改转到决策树的构造过程中,对新产生的节点再重复进行上面的操作。

4 算法的描述

由上面的讨论,综合得到算法描述如下:

(1) 令 U 为所有对象的集合,A 为所有条件属性的集合,创建一个节点 N。

(2) 如果 U 中所有对象都属于同一个决策类 C,则返回 N 作为叶子节点,用类 C 的标志来标记;否则

(3) 计算 A 中所有的属性的信息增益率,选择具有最大

增益率的属性 a,将对象集 U 不完全划分为 k 类:$U_1, U_2, \ldots, U_k$,相应得到 k 个子节点 $N_1, N_2, \ldots, N_k$,如果有第 U_i 中也即第 i 个子节点中含有在 a 属性取值不确定的对象,则按照前面提到的两种补充方法进行修改,直到所有的子节点含有的对象在 a 的取值都已经确定;

(4) 对于所有新得到的子节点 N_i,令 $A = A\backslash\{a\}$(即从属性集 A 中去掉上一步中具有最大增益率的属性 a),$U = U_i$, $N = N_i$, 重复(2) ~ (3) 的过程;

(5) 当所有的节点都成为叶子节点或 A 为空集时,决策树构造完毕;

(6) 由得到的决策树,从根结点到叶子节点,由上而下的每条路径提取所有的决策规则;

(7) 如果决策表中还有 * 没有确定,再令 U 为所有对象的集合,利用相似度来确定。

下面我们用一个简单的例子来说明算法的具体操作过程。

5 例子详细介绍

我们考虑如表 1 所表示的决策表:

表 1

	条件属性			决策属性
	a	b	c	d
1	H	N	L	0
2	L	*	S	1
3	H	Y	L	0
4	*	Y	N	0
5	L	N	L	2
6	N	N	L	2
7	*	Y	N	0
8	N	*	S	1
9	L	Y	L	0
10	N	N	*	1

在表 1 中, * 表示对象的属性值遗失。表 1 是一个不完全决策表,含有 10 个对象 $U = \{1,2,3,4,5,6,7,8,9,10\}$;3 个条件属性 $A = \{a,b,c\}$;决策属性 d 有 3 个取值,将对象 U 划分成 3 个等价类:$C_0 = \{1,3,4,7,9\}$, $C_1 = \{2,8,10\}$, $C_2 = \{5,6\}$; A 中每个条件属性将 U 不完全划分为:$U_a = \{\{1,3,4^*,7^*\}, \{6,8,10,4^*,7^*\}, \{2,5,9,4^*,7^*\}\}$; $U_b = \{\{1,5,6,10,2^*,8^*\}, \{3,4,7,9,2^*,8^*\}\}$; $U_c = \{\{1,3,5,9,6,10^*\}, \{4,7,10^*\}, \{2,8,10^*\}\}$,其中对象加 * 表示该对象的属性取值遗失。因为 U 中的对象不是属于同一个决策类,故需要选择测试的条件属性,即信息增益率最大的条件属性,先由公式(1) 计算信息熵 $I(U)$:

$$I(U) = -\left(\frac{5}{10}\log_2\frac{5}{10} + \frac{3}{10}\log_2\frac{3}{10} + \frac{2}{10}\log_2\frac{2}{10}\right) = 1.485475$$

再由公式(2) 计算条件属性 a 为节点所需的期望信息:

$$E(a) = \sum_{i=1}^{m}\frac{|U_i|}{|U|}I(U_i) = \frac{3}{10}\times\left(-\left(\frac{2}{5}\log_2\frac{2}{5} + \frac{2}{5}\log_2\frac{2}{5} + \frac{1}{5}\log_2\frac{1}{5}\right)\right) +$$

$$\frac{3}{10}\times\left(-\left(\frac{3}{5}\log_2\frac{3}{5} + \frac{1}{5}\log_2\frac{1}{5} + \frac{1}{5}\log_2\frac{1}{5}\right)\right) = 0.867864$$

从而由公式(5) 得到 a 的信息增益率为:

$$GR(a) = \frac{I(U) - E(a)}{IV(a)} = \frac{1.485475 - 0.867864}{-\left(\frac{2}{10}\log_2\frac{2}{10} + \frac{3}{10}\log_2\frac{3}{10} + \frac{3}{10}\log_2\frac{3}{10}\right)} = \frac{0.617611}{1.506565} = 0.4099465$$

类似的,我们再计算 b,c 为节点所需的期望信息分别为:

$E(b) = 1.084311$ $E(c) = 0.9962406$

对应信息增益率分别为:

$GR(b) = 0.3793361$ $GR(c) = 0.335657$

因为 a 的信息增益率最大,故选取 a 属性为根结点的测试属性,将对象分成不完全的三类:$U_1 = \{1,3,4^*,7^*\}$, $U_2 = \{6,8,10,4^*,7^*\}$, $U_3 = \{2,5,9,4^*,7^*\}$,对应得到决策树的三个子节点 N_1, N_2, N_3,得到的决策树如图 1 所示:

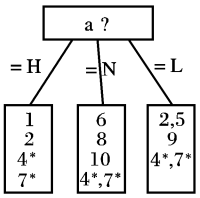

图 1

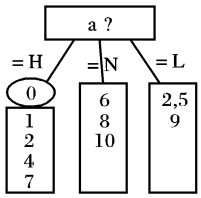

图 2

对于新产生的三个节点,因为含有带 * 的对象,故利用 3 小节中提到的方法(1) 和(2) 进行修改,注意到 N_1 节点所含的对象 U_1 都属于第 C_0 类,故按照方法(1) 修改,得到 $a(4) = a(7) = H$,同时 N_1 节点成为叶节点,用 C_0 类的类标志 0 标记。修改之后的决策树如图 2 所示。改变 U 和 A 重复进行上面的操作,最后我们得到的决策树如图 3 所示。决策表中还有不确定值,如果希望修改,再按照方法(2) 修改,最后修改完的决策表见表 2。

表 2

	条件属性			决策属性
	a	b	c	d
1	H	N	L	0
2	L	N	S	1
3	H	Y	L	0
4	H	Y	N	0
5	L	N	L	2
6	N	N	L	2
7	H	Y	N	0
8	N	N	S	1
9	L	Y	L	0
10	N	N	S	1

从决策树(图 3) 上提取决策规则如下:

(a) $a = H => d = 0$;

(b) $a = N \& c = L => d = 1$;

(c) $a = N \& c = S => d = 2$;

(d) $a = N \& b = Y => d = 0$;

(e) $a = L \& b = N \& c = S => d = 1$

(*f*) $a = L \& b = N \& c = L => d = 2$

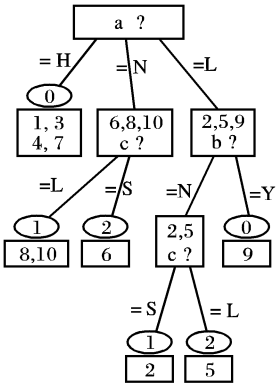

图 3

6 结论

正确填充遗失数据是数据挖掘预处理的一个难点问题，从不完全的决策表中提取决策规则也一直是数据挖掘技术的重要任务之一。本文提出的基于决策树的算法，有效地解决了这两个问题，在构造决策树时使用信息增益率作为选择测试属性的准则，遗失数据的填充考虑了数据之间的关系，具有更好的可行性。

参考文献

[1] HONG TP, Tseng LH, Wang SL. Leatning rules from incomplete training examples by rough sets[J]. Expert Systems with Application, 2002, (22): 258 – 293.

[2] Han JW, Kamber M. 数据挖掘概念与技术[M]. 范明, 孟小峰, 译. 北京: 机械工业出版社, 2001. 70 – 74, 185 – 196.

[3] Quinlan JR. Decision trees and decisionmaking[J]. IEEE Transaction on Systems, Man, and Cybernetics, 1990, 20(2): 339 – 346.

[4] 郭景峰, 米浦波, 刘国华. 基于决策树的数据遗失值填充方法的研究[J]. 计算机工程与科学, 2002, 24(5).

[5] 黄欣, 杨杰, 叶晨洲. 基于复合式衡量准则的决策树生成算法[J]. 上海交通大学学报, 2000, 43(12).

[6] 赵东卫, 李旗号. 粗集在决策树优化中的应用[J]. 系统工程学报, 2001, 16(4).

[7] 张文修, 等. 粗糙集理论于方法[M]. 北京: 科学出版社, 2001. 206 – 211.