决策树算法综述

谢金梅,王艳妮

(中国地质大学 资源学院,湖北 武汉 430074)

摘 要:分类和预测是数据挖掘的主要方法之一,用于提取描述重要数据类的模型或预测未来的数据趋势。决策树方法是一种应用广泛的分类方法。在介绍决策树方法的基本思想与实现步骤的基础上,重点介绍了ID3算法、C4.5算法、χ^2统计算法和SPRINT算法 。

关键词:数据挖掘;分类;决策树

中图分类号:TP312 文献标识码:A 文章编号:1672-7800(2008)11-0083-02

1 决策树分类方法的思想

决策树分类是一种从无次序、无规则的训练样本集中推理出决策树表示形式的分类规则的方法。它采用自顶向下的方法,在决策树的内部结点进行属性值的比较并根据不同的属性值判断从该结点向下的分支,在决策树的叶结点得到结论。所以从根结点到任一个叶结点所形成的一条路径就构成了一条分类规则,其中路径中的属性值偶对就构成了分类规则条件部分(IF部分)中的一个合取项,叶结点所标记的类别就构成了规则的结论内容(THEN部分)。如图1所示,决策树的生成过程包括两个阶段:树构造和树剪枝。

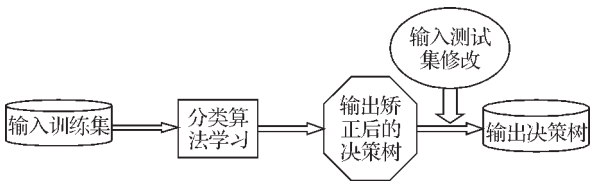

图1 决策树的生成过程

1.1 树构造

决策树采用自顶向下的递归方式:从根节点开始在每个节点上按照给定标准选择测试属性,然后按照相应属性的所有可能取值向下建立分枝、划分训练样本,直到一个节点上的所有样本都被划分到同一个类,或者某一节点中的样本数量低于给定值时为止。这一阶段最关键的操作是在树的节点上选择最佳测试属性,该属性可以将训练样本进行最好的划分。最佳测试属性的选择标准有信息增益、基尼指数以及基于距离的划分等。

1.2 树剪枝

构造过程得到的并不是最简单、紧凑的决策树,因为许多分枝反映的可能是训练数据中的噪声或孤立点。树剪枝过程试图检测和去掉这种分枝,以提高对未知数据集进行分类时的准确性。树剪枝方法主要有先剪枝和后剪枝。树剪枝方法的剪枝标准有最小描述长度(MDL)和最小期望错误率等。前者对决策树进行二进位编码, 最佳剪枝树就是编码所需二进位最少的树; 后者计算某节点上的子树被剪枝后出现的期望错误率,由此判断是否剪枝。

2 主要决策树算法

2.1 ID3算法

当前最有影响的决策树算法是Quinlan于1986年提出的ID3和1993年提出的C4.5算法。ID3算法是较早出现也是最著名的决策树归纳算法。C4.5是ID3的改进算法,不仅可以处理离散值属性,还能处理连续值属性。

ID3算法是将信息增益分析技术和基于多维数据分析的方法集成在一起,删除信息量较少的属性,收集信息量较多的属性,常用于概念分析。设C是样本中类的数目,S是样本数,$p(s,j)$表示样本S中样本属于第j类的概率,也即$p(s,j)=s_j/S$,s_j是样本S中属于类j 的样本数。因此,对于一个给定的样本分类所需的期望信息增益是:

$$Info(S)=-\sum_{j=1}^{c} p(S,j)\log_2 P(S,j)$$

具有值$\{a_1,a_2,\cdots a_k\}$的属性T可以将S划分为子集$\{S_1,S_2,\cdots S_k\}$,其中S_j包括样本S中T的值为a_j的那些样本。设S_j包括类C_i的S_{ij}个样本,根据T的这种划分的期望信息称作T的熵。其加权平均为:$E(T)=\sum_{i=1}^{k}\frac{|S_i|}{|S|}log_2\frac{|S_i|}{|S|}$

作者简介:谢金梅(1982~),女,新疆博乐人,中国地质大学(武汉)硕士研究生,研究方向为地学信息工程;王艳妮(1984~),女,陕西韩城人,中国地质大学(武汉)硕士研究生,研究方向为地球探测与信息技术。

T上划分获得的信息增益定义为：$Gain(S,T)=Inf o(S)-E(T)$

ID3算法优化：通过加权和增加属性重要度，加强属性的标注，降低了非重要属性的标注，把“加权和”转换为权与属性重要度相加的“新加权和”。生成决策树时，数据少的数据元组不会被淹没，最终使决策树减少“大数据掩盖小数据”现象的发生。利用属性重要度5，类别条件熵：

$$H(C|V)=-\left(\left(\sum p(Vi)+s\right)\sum p(C_j|V_i)log_2 p(C_j|V_i)\right)$$

信息增益的公式变为：

$$I(C,V)=H(C)-H(C|V)=inf\ o(T)-inf\ ov(T)=gain(v)$$

属性重要度取值在[0,1]之间，其大小由训练数据集数据计算给出。

2.2 C4.5算法

在ID3算法中，具有最大信息增益的属性被选为分裂属性。显然，ID3算法偏袒具有较多值的属性，因而可能导致过度拟合。因此，在C4.5算法中采用信息增益比率来代替信息增益。

增益比率为：$GainRatio(S,T)=\dfrac{Gain(S,T)}{Inf o\left(\dfrac{|S_1|}{|S|},\dfrac{|S_2|}{|S|},\cdots,\dfrac{|S_k|}{|S|}\right)}$

为了达到最佳分裂的目的，C4.5先计算每个属性的增益，然后仅对那些高于信息增益平均值的属性应用增益比率进行测试。增益比率最大的属性应当首选为分裂属性。

2.3 χ^2统计算法

统计算法是Hart和Mingers设计的另外一种属性区分度量方法。这是一种传统的统计方法，在一个可能性表中，它度量任意两个变量之间的联系。通过比较变量的观察频数和期望频数来确定两个变量之间是否有联系，其结果近似服从χ^2分布，值越大则表明联系越强。基本的方程如下：$\chi^2=\sum\sum\dfrac{(x_{ij}-E_{ij})^2}{E_{ij}}$

在上面的方程中，$E_{ij}=x_i x_j/N$，即可能性表中每一项的期望值。

2.4 并行决策树算法

并行决策树将一个数据集随机的分成几个部分，各部分数据分别用不同的处理器学习，选出各自最好的属性，然后通过比较，综合得出最好的属性，以它为基准建树。对每个新节点，递归地重复上述过程，直到所有的节点都展开成树叶。图2是将一个数据集分成3个子数据集时的学习流程图。

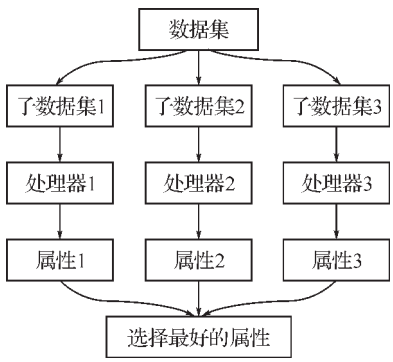

图2 一分三学习流程

并行决策树构建的主要步骤如下：

(1)根据决策树扩展策略，随机选择一节点作为当前的节点，并将开始时当前的节点选为根节点；

(2)当前节点的数据集分成几个子集，各个数据子集分别用一个处理器(学习器)学习；

(3)各处理器分别用ID3，C4.5，χ^2统计等方法生成各自最好的属性；

(4)选择一个最好的属性作为分裂属性；

(5)依靠期望树分枝因子，创建本节点的孩子节点；

(6)对每个子节点，重复(1)~(5)，直到没有节点能被用来扩展树；

(7)用EBP方法修剪最终的树。

2.5 SPRINT算法基本思想

由于上述方法存在决策树的过适应问题，为提高准确率，我们需要消除训练集中的异常和噪声。一般有两种方法：先剪枝法(Public 算法)和后剪枝法(Sprint 算法)。SPRINT剪枝采用了最小描述长度原则。

2.5.1 SPRINT 算法

分割指数(Spihing Index)被用来评估属性上分割规则的优劣程度。Gine参数已经被证明能够有效地搜索最佳分割点。假设一个训练集s有n条记录，它们分别属于m个不同的类，则集合s的分割指数gini定义如下：

$$gini(S)=1-\sum_{j=1}^{c}p_j^2 \quad (1)$$

如果使用分割规则cond将s划分为s1和s2两个子集，则该规则的评估值记为gini(s，cond)，定义如下所示：

$$gini^D(S,cond)=\frac{n_1}{n}gini(S_1)+\frac{n_2}{n}gini(S_2) \quad (2)$$

对于一个数值型属性A，它的分裂形式为$A\leqslant v$或者$A>V$两个部分，所以，我们可以先对数值型属性进行排序。假设排序后的结果是$v_1,v_2....v_i$，因为分裂只发生在两个接点间，所以有n一1种可能性，通常取中点$(v_i+v_i+1)/2$作为分裂点，从小到大依次取不同的分裂点，取gini值最小的点作为最佳分裂点。这种寻找最佳分裂点的方法能够找到最精确的分裂点。但是对于数值型属性，首先要对整个训练集进行预排序，然后将每两个结点之间的中点值都作为分裂点来计算gini值，工作量很大，特别是对于超大数据集，当属性含有大量的不同取值时，效率非常低。

2.5.2 改进的SPRINT 算法

可以证明若$f(x)$是凸函数，则$f(x)$在一个区间内的极小值只可能出现在区间的边界点处。这样就只需要计算区间边界的gini值，就可以得到该纯区间的极小值，减小计算量。

对数据集S的某一个数值型属性分裂：

(1)由于数值型属性的分类一般服从高斯分布，因此用等宽直方图方法将属性值分为q个区间，同时构造区间方图列表。区间方图列表的字段有区间的左边界、右边界的值和该区间中各个类的记录数。

(2)对每一个区间计算gini值，并找出最小值$gini_{low}$。对于区

Java反射机制探究

尹松强,傅 鹂

(重庆大学,重庆 400444)

摘 要:反射是提高Java程序的灵活性和可扩展性至关重要的技术,它使得Java软件系统具有自适应能力,实现程序的动态演进。阐述了Java 反射机制的原理和工作过程,并通过应用实例证明Java反射机制的强大威力和使用它应注意之处。

关键词:Java;反射;类替换;性能

中图分类号:TP312 文献标识码:A 文章编号:1672-7800(2008)11-0085-03

0 引言

反射(reflection)的概念是由Smith 在1982 年首次提出的,主要是指程序可以访问、检测和修改它本身状态或行为的一种能力。这一概念的提出很快引发了计算机科学领域关于应用反射性的研究。它首先被程序语言的设计领域所采用, 并在Lisp和面向对象方面取得了成绩。最近,反射机制也被应用到了视窗系统、操作系统和文件系统中。

1 Java反射机制原理

反射是Java被认为是动态语言的一个非常重要的特性,它允许动态发现和绑定类、方法、字段,以及所有其他的由语言所产生的元素。也就是说,这类应用通过采用某种机制来实现对自己行为的描述和监测, 并能根据自身行为的状态和结果,调整或修改应用所描述行为的状态和相关的语义。

当前主流的计算反射原型系统大多是建立在元对象协议

间$[v_b,v_1]$,则计算区间右边界处的$gini^D(S,a\leqslant vl)$值;对于非闭区间,建立区间属性表,再对区间进行排序,然后精确计算在该区间最小gini值。

(3)用最小gini值$gini_{low}$对属性表进行分裂。改进的SPIRNT算法利用了gini函数在闭区间上是凸函数和数值型属性一般服从高斯分布的特点进行了一定的改进,在一定程度上加快了区间中结点值的选取速度,从而大大减少了算法的时间代价和空间代价。但是改进的SPRINT算法不是十全十美的,因为区间划分在很大程度上影响着分裂点的划分,从而严重影响着数据挖掘中决策树的生成速度,所以如何有效的划分区间值得大家继续探讨的问题。

3 结束语

虽然基于决策树的学习算法具有建立速度快、精度高、可以生成可理解的规则、计算量不是很大、可以处理连续值和离散值属性、可以清晰显示哪些属性比较重要等优势,但决策树技术又是一种“贪心”搜索,因此使用决策树方法时,还会遇到一些数据准备和数据表示方面的问题,如:不了解数据细节、数据翻译不准确、不考虑字段间关系、数据表示需要加额外的数据、数据噪音消除、欺骗性数据等,这些均会影响到最终生成决策树和规则提取准确性和实用性,也就是说其应用范围是有一定的局限性的。决策树技术应继续在寻找更好的简化决策树的方法、研究产生决策树的训练和检验数据的大小及特性与决策树特性之间的关系、不确定环境下决策树研究、决策树时间复杂度与准确性之间的矛盾以及决策树技术的软件实现等方面不断完善。

参考文献:

[1] J Han,M Kamber.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.
[2] 史忠植.知识发现[M].北京:清华大学出版社,2002.
[3] Quinlan J R.Induction of decision tree.Machine Learning[J].1986(1).
[4] 魏晓云.决策树分类方法研究[J].计算机系统应用,2007(9).
[5] 王玉珍.基于数据挖掘的决策树方法分析[J].电脑开发与应用.2007(5).
[6] 王名扬.基于数据挖掘的决策树生成与剪枝方法[J].计算机工程与科学,2005(11).

(责任编辑:赵 峰)

作者简介:尹松强(1980~),男,河南开封人,重庆大学硕士研究生,研究方向为网络与数据库。