决策树算法的研究与改进

冯 少 荣*
(华南理工大学计算机科学与工程学院,广东 广州 510641)

摘要: 决策树是数据挖掘中重要的分类方法,本文在研究和比较几种经典的决策树算法基础上,提出了一种改进的决策树算法:基于度量的决策树(MBDT).这种决策树实际上是把线性分类器和决策树结合在一起.实验证明,用该方法构造的决策树能有效地减少决策树的层数,从而提高决策树的分类效率.通过 MBDT 分类实验,验证了上面结论的正确性和有效性.

关键词: 决策树;度量;ID3 算法;熵
中图分类号: TP 18 文献标识码: A 文章编号: 0438-0479(2007)04-0496-05

决策树(Decision Tree)是用于分类和预测的主要技术,它着眼于从一组无规则的事例推理出决策树表示形式的分类规则,采用自顶向下的递归方式,在决策树的内部节点进行属性值的比较,并根据不同属性判断从该节点向下分支,在决策树的叶节点得到结论.因此,从根节点到叶节点就对应着一条合理规则,整棵树就对应着一组表达式规则.基于决策树算法的一个最大的优点是它在学习过程中不需要使用者了解很多背景知识,只要训练事例能够用属性即结论的方式表达出来,就能使用该算法进行学习.

基于决策树的分类模型有如下几个特点:

(1)决策树方法结构简单,便于理解;(2)决策树模型效率高,对训练集数据量较大的情况较为适合;(3)决策树方法通常不需要接受训练集数据外的知识;(4)决策树方法具有较高的分类精确度.

近年来,决策树方法在机器学习、知识发现等领域得到了广泛应用.到目前为止国内外的研究人员先后提出了十几种与决策树相关的分类方法[1-11],对分类问题给出了程度不同的解决方案,但都存在一定程度的不足.

1 几种经典的决策树算法简述

1.1 ID3(Iterative Dichotomizer 3)算法[2]

ID3 是一种经典的决策树算法,它从根节点开始,根节点被赋予一个最好的属性.随后对该属性的每个取值都生成相应的分支,在每个分支上又生成新的节点.对于最好的属性的选择标准,ID3 采用基于信息熵定义的信息增益来选择内节点的测试属性,熵(Entropy)刻画了任意样本集的纯度.

设 S 是 n 个数据样本的集合,将样本集划分为 c 个不同的类 $C_i(i = 1, 2, \cdots, c)$,每个类 C_i 含有的样本数目为 n_i,则 S 划分为 c 个类的信息熵或期望信息,有

$$E(S) = \sum_{i=1}^{c} p_i \log_2(p_i),$$

其中 p_i 为 S 中样本属于第 i 类 C_i 的概率,即 $p_i = \frac{n_i}{n}$.

假设属性 A 的所有不同值的集合为 X_A,S_v 是 S 中属性 A 的值为 v 的样本子集,即 $S_v = \{s \in S \mid A(s) = v\}$,在选择属性 A 后的每一个分支节点上,对该节点的样本集 S_v 分类的熵为 $E(S_v)$.选择 A 导致的期望熵定义为每个子集 S_v 的熵的加权和,权值为属于 S_v 的样本占原始样本 S 的比例 $\frac{|S_v|}{|S|}$,即期望熵为

$$E(S, A) = \sum_{v \in X_A} \frac{|S_v|}{|S|} E(S_v),$$

其中,$E(S_v)$ 是将 S_v 中的样本划分到 c 个类的信息熵.属性 A 相对样本集合 S 的信息增益 $\mathrm{Gain}(S, A)$ 定义为

$$\mathrm{Gain}(S, A) = E(S) - E(S, A),$$

其中 $\mathrm{Gain}(S, A)$ 是指因知道属性 A 的值后导致的熵的期望压缩.$\mathrm{Gain}(S, A)$ 越大,说明选择测试属性 A 对分类提供的信息越多.ID3 算法就是将每个节点选择信息增益 $\mathrm{Gain}(S, A)$ 最大的属性作为测试属性.

ID3 算法的局限是它的属性只能取离散值,为了使决策树能应用于连续属性值,Quinlan 给出了 ID3 的一个扩展算法,即 C4.5 算法.

1.2 C4.5 算法[4]

收稿日期: 2006-06-01
基金项目: 福建省自然科学基金(A0310008),福建省高新技术研究开放计划重点项目(2003H043)资助
* 现工作单位: 厦门大学计算机科学系
Email: shaorong@xmu.edu.cn

C4.5 算法是 ID3 的改进, 其中属性的选择依据同 ID3. 它对于实值变量的处理与下节论述的 CART(Classification And Regression Trees) 算法一致, 采用多重分支. C4.5 算法能实现基于规则的剪枝. 因为算法生成的每个叶子都和一条规则相关联, 这个规则可以从树的根节点直到叶节点的路径上以逻辑合取式的形式读出.

1.3 CART 算法[1]

决策树的分类过程就是把训练集划分为越来越小的子集的过程. 理想的结果是决策树的叶子节点的样本都有同类标记. 如果是这样, 显然决策树的分支应该停止了, 因为所有的类别已经被分开了. 但是, 一般情况下, 很难一步就达到目标, 所以, 如果不止一步才能结束的话, 这个分类的过程就是一个递归树的生长过程, CART 是仅有的一种通用的树生长算法.

1) 节点属性选择准则

CART 的节点选择准则是使节点的不纯度尽可能小, 即使得不纯度尽可能大地下降. 而不纯度(impurity) 是一个和纯度相对的概念. 度量一个节点的不纯度比度量纯度更有利于分类, 所以使用不纯度作为指标. 不纯度的度量不是固定的, 主要有以下几种:

i) 熵不纯度的计算公式为

$$\mathrm{Imp}(N) = -\sum_j P(\omega_j)\log_2 P(\omega_j) \tag{1}$$

其中 $P(\omega_j)$ 是节点 N 处属于 ω_j 类样本数占总样本数的比例. 显然, 如果所有的样本都是同一类别的, 则不纯度为零, 否则就是一个大于零的正值.

ii) Gini 不纯度也被称作方差不纯度. 一个两类问题的方差不纯度定义为

$$\mathrm{Imp}(N) = P(\omega_1)P(\omega_2) \tag{2}$$

很明显, 这个值和两类分布的总体分布方差有关. 推广式(2) 用于多类分类问题时它的值由下式得到

$$\mathrm{Imp}(N) = \sum_{i\neq j} P(\omega_i)P(\omega_j) = 1 - \sum_j P^2(\omega_j) \tag{3}$$

这就是 Gini 不纯度.

iii) 误分类不纯度

$$\mathrm{Imp}(N) = 1 - \max P(\omega_j) \tag{4}$$

它用来衡量节点的样本分类误差的最小概率. 这个指标的峰值特性是最好的, 但它的缺陷是导数不连续, 因而在连续参数空间搜索最大值时会出现问题.

如果通过不纯度来选择属性, 与 ID3 中的最大化信息增益类似. 要使得不纯度下降最大, 不纯度下降可以使用下式计算

$$\Delta\mathrm{Imp}(N) = \mathrm{Imp}(N) - P_L\mathrm{Imp}(N_L) - P_R\mathrm{Imp}(N_R) \tag{5}$$

其中 N_L 和 N_R 是左右子节点, $\mathrm{Imp}(N_L)$ 和 $\mathrm{Imp}(N_R)$ 是相应的不纯度, 这个式子只能生成一颗二叉树. 一个用于多类问题的简单推广是

$$\Delta\mathrm{Imp}(s) = \mathrm{Imp}(N) - \sum_{k=1}^{M} p_k\mathrm{Imp}(N_k) \tag{6}$$

其中 p_k 是分支到节点 N_k 的训练样本占的比例, 且满足 $\sum_{k=1}^{M} p_k = 1$. 然而, 这个式子存在缺陷, 如果 M 越大, 显然 $\Delta\mathrm{Imp}(N)$ 也越大, 但是这个时候的分类未必是更有意义的, 譬如随机分布. 因此, 对式(6) 的一个改进是

$$\Delta\mathrm{Imp}M(s) = \frac{\Delta\mathrm{Imp}(s)}{-\sum_{k=1}^{M} P_k\log_2 p_k} \tag{7}$$

这个公式实际上是式(6) 的归一化.

2) 分支停止准则

一般地, 总存在一些样本不能从另一类样本中分离出来, 这就需要判断什么时候决策树的生长应该停止. 如果所有的节点都达到最小的不纯度为止, 那么数据很可能会被“过拟合”. 极端情况下, 节点只对应单一样本, 决策树就变成一个查找表, 这样对有较大噪声的训练样本的推广性能就不会很好. 相反, 如果决策树停止的太早, 对训练样本的误差就不够小, 因而分类能力就很差.

一种判断停止分支的方法是验证和交叉验证技术, 即选择一部分样本进行训练, 然后用剩余的样本作测试验证, 直到对于验证集的分类误差最小化为止. 另一种方法是预先设定一个不纯度下降差的阈值. 当候选分支使得节点的不纯度下降差小于这个阈值, 即 $\max_s \Delta\mathrm{Imp}M(s) < \mathrm{th}$ 时, 停止分支.

3) 剪枝(Pruning)

剪枝也是一种停止分支的方法. 它的前提是决策树充分的生长, 使得叶节点有最小的不纯度为止. 然后, 对所有相邻的成对叶节点, 考虑是否消去它们. 标准是如果消去它们使得不纯度的增长很小, 就执行消去. 很明显, 剪枝的过程恰好是节点分支的逆过程. 当然, 从叶节点开始剪枝并非必要. 基于不纯度增长的判断标准, 可以从任何节点开始剪枝.

剪枝技术的一个缺陷是, 如果样本数很大的话, 则计算量代价会非常高, 甚至无法实现. 不过, 在一般的情况下, 剪枝的方法优于上一节提到的分支停止方法.

2 改进的决策树算法(Metric Based Decision Tree, MBDT)

对任何数量的训练集, 总是能找到相应的多个线性判别函数把它分类, 但是这样生成的树的深度可能太大. 因为, 虽然使用了最好的特征进行分类, 但还是可能存在一些特征对分类很有用, 尽管不是像最好的特征那样有用, 却没有用到. 一个直觉是: 有些特征对某些类别有效, 但是对另外一些则无效, 甚至可能有副作用, 如果能把这些特征选择出来, 一次就能最大限度地把多个类别分开. MBDT 正是基于这个直觉. MBDT 通过在每个子集上选择最能有效分类的那些特征使用马氏距离进行分类. 如果某个子集无法有效分类(通过阈值判断), 就选择最好的一个进行分类. 由于事先需要有标签的分类训练集, 所以这是有监督的算法.

2.1 MBDT 的度量方法

度量数据相似性的线性方法有多种, 常用的有欧氏距离、棋盘距离、马氏距离和切比雪夫距离等. 马氏距离的特点是对于比例尺的变换有不变性.

令 $y = Ax$, 那么向量 x_1, x_2 和 m_x 之间的距离与经过变换后的 y_1, y_2 和 m_y 之间的距离显然不同, 甚至会出现这样的情况: 即 $\| x_1 - m_x \|_G > \| x_2 - m_x \|_G$, 但是 $\| y_1 - m_y \|_G < \| y_2 - m_y \|_G$. 其中 G 表示某个范式. 这样的话, 数据之间的相近程度就不是客观的, 我们无法度量. 马氏距离具有在改变比例尺的情况下, 保持距离尺度的特性. 马氏距离为

$$\| x - m \|_M = (x - m)^T C^{-1} (x - m) \qquad (8)$$

在马氏距离尺度下, 选择合适的协方差矩阵, 可以调整分类器(也就是对 x 进行变换), 使得样本可以聚类为任何一种形式的超椭球体, 从而为使用距离判据提供了基础.

2.2 MBDT 的算法

MBDT 算法也是递归的. 它的分支准测采用阈值方式. 令 $T = \{t_i\}$, $1 \leqslant i \leqslant c$ 表示样本集合, 其中 c 是样本类别个数. 令 $A = \{a_i\}$, $1 \leqslant i \leqslant m$ 表示特征空间, 其中 m 是属性的个数, 则 $B = \{b \mid b \subseteq A\}$ 就是 A 的幂空间. 令 β_c 表示误分类阈值, β_e 表示交叉误分类阈值.

i) 对于一个超集 $C \subseteq T$, C 包含几个类别的样本. 对于一个属性集合 $b \in B$, 如果 C 中的样本在这个属性集上的取值 $x = (x_1, \cdots, x_n)$, $x_i \in b$, $n = \mid b \mid$ 与某个类的典型模式的马氏距离最小, 就把样本归入这个类. 我们的目标是选择一个属性集合 $b_{best} \in B$, 使得 C 中的样本尽可能多地被正确分类. 在实践中, 总存在一些数据无法正确分类, 所以如果误分类的比例小于 β_c 就判定是尽可能地被正确分类了. 如果类 t_i 的样本被误分类到 t_j 的比例大于 β_e, t_i 和 t_j 就被归入同一类. 等待下一层再继续分类.

ii) 如果找不到一个属性集合满足 i), 就选择最好的情况进行分类.

iii) 如果所有的样本被分类或者无法继续进行分类, 那么这个过程结束.

需要强调的是, 这个过程得到的最终结果未必是最优的, 特别是在 i) 中满足条件的分类方式不止一个的时候, 所以, 如果要得到最优的结果, 需要搜索整个 B 空间中的不同分类情况. 另外, 根据试验的结果, 如果只用一个属性进行分类的话, 欧氏距离的效果比马氏距离更好. 最后, 对 MBDT 一个改进可能会使得分类的效果更好, 即把 MBDT 和别的方法结合起来, 比如说 CART. MBDT 能快速地把多个类别分开, 所以可以在使用 MBDT 后, 再用 CART 进行分类, 因为样本类别的数目已经相当少了.

3 MBDT 的实验结果

表 1 给出了实验中所用到的 5 种、4 个特征的 20 组数据, 最终生成的决策树为 3, 如图 1 所示.

表 1 5 种、4 个特征的 20 组数据

Tab. 1 Five class, twenty group data of possessing four attributes

项目	对比度	一致性	相关性	熵
T1	38.13023	2.36818	-0.00001	20.76877
	37.04579	2.81426	0	21.68249
	25.32734	2.41561	-0.00001	18.1081
	49.81108	3.91428	0	24.3985
T2	3.05189	1.09292	-0.01228	4.67713
	2.98452	1.13592	-0.0104l	4.7945
	2.67046	1.05752	-0.02126	4.16217
	2.61475	1.06905	-0.01702	4.5049
T3	155.899	1.44947	-0.00027	13.14602
	171.7583	1.89651	-0.00018	11.89057
	150.4348	1.53434	-0.00026	12.47183
	120.5233	1.23407	-0.00057	11.78203
T4	206.7022	1.25251	-0.00005	19.74323
	85.85167	1.09008	-0.0007	9.68642
	120.5504	1.07904	-0.00033	12.31361
	92.64148	1.03967	-0.00059	11.06669
T5	104.5755	2.66829	0	33.63455
	51.01725	1.71086	-0.00005	21.71022
	104.031	1.93512	-0.00001	32.43497
	94.98301	2.27124	0	32.18267

为了使用验证和交叉验证方法, 实验中把每种的 5 组数据分成 A 和 B 两部分, 分别用来训练和验证. 所以, 最后会产生四类结果, 即: A 训练 A 验证、A 训练 B

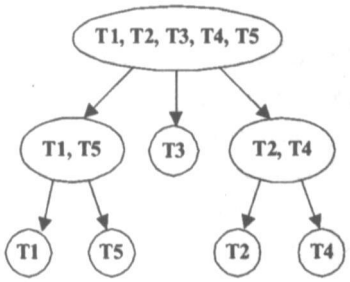

图 1　MBDT 在测试数据集上生成的决策树

Fig. 1　Built decision trees based on testing data set

验证, B 训练 A 验证, B 训练 B 验证. 每一个验证结果都是一个矩阵, 矩阵对角线上的元素表示被正确分类的样本数, 矩阵$[i,j](i \neq j)$位置上的元素表示第 i 类被误分为第 j 类.

表 2 给出了第 1 层的分类结果. 在第 1 层分成 3 子类: T1 和 T5、T3、T2 和 T4. 表 3 是第 2 层对 T2 和 T4 两种分类的结果. 表 4 是第 2 层对 T1 和 T5 两种分类的结果.

通过在不同数据集上分别运行 ID3、C4.5、MBDT 算法, 得到的实验数据如表 5. 由表 5 我们可以得到, 在分类性能上, MBDT 算法较 ID3 及 C4.5 有明显提高. 图 2 是它们运行结果的性能指标比较.

表 2　使用对比度、一致性和熵的分类结果

Tab. 2　Classified result of making use of contrast, consistency and entropy

	A 组					B 组				
A 组	5	0	0	0	0	5	0	0	0	0
	0	4	0	1	0	0	5	0	0	0
	0	0	5	0	0	0	0	5	0	0
	0	0	0	5	0	0	0	0	5	0
	0	0	0	0	5	0	0	0	0	5
B 组	5	0	0	0	0	4	0	0	0	1
	0	5	0	0	0	0	4	0	1	0
	0	0	5	0	0	0	0	5	0	0
	0	0	0	5	0	0	0	0	5	0
	0	0	0	0	5	0	0	0	0	5

表 3　使用对比度和一致性的分类结果(T2 与 T4)

Tab. 3　Classified result of making use of contrast and consistency(T2 and T4)

	A 组	A 组	B 组	B 组
A 组	5	0	5	0
A 组	0	5	0	5
B 组	5	0	5	0
B 组	0	5	0	5

表 4　使用对比度和一致性的分类结果(T1 与 T5)

Tab. 4　Classified result of making use of contrast and consistency(T1 and T5)

	A 组	A 组	B 组	B 组
A 组	5	0	5	0
A 组	0	5	0	5
B 组	5	0	4	1
B 组	0	5	0	5

表 5　改进算法构造的决策树与传统 ID3 及 C4.5 算法构造的决策树比较

Tab. 5　Comparing built decision trees based on improvement algorithm with built decision trees based on tradition ID3 and C4.5

事例数	ID3		C4.5		改进算法	
	节点数(个)	错误率(%)	节点数(个)	错误率(%)	节点数(个)	错误率(%)
100	10	0.9	8	0.0	10	0.7
100	15	0.0	8	0.7	10	0.0
200	20	3.0	15	0.5	12	0.6
200	25	0.6	16	2.0	15	1.4
300	30	1.8	25	1.5	20	1.0

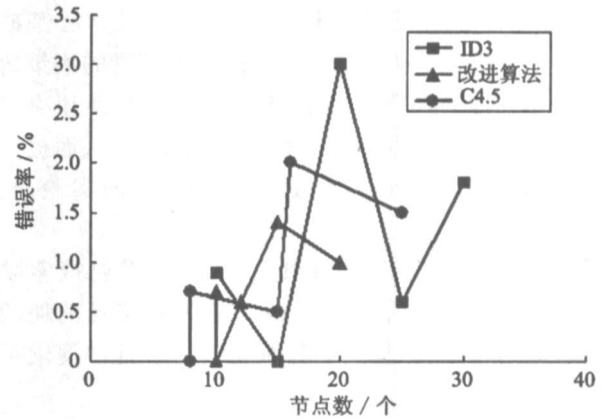

图 2　改进算法构造的决策树与传统 ID3 及 C4.5 算法构造的决策树比较

Fig. 2　Comparing built decision trees based on improvement algorithm with built decision trees based on tradition ID3 and C4.5

从图 2 和表 5 可以看出改进算法构造的决策树明显优于传统 ID3 及 C4.5 算法构造的决策树.

3.1　MBDT 实验结果的分析

通过上面的实验结果我们能够看到, MBDT 生成了一颗三层决策树, 也就是说查询两次就能得到要求的分类. 在最终的分类结果中, 只有一组数据被分类错误. 观察被误分类的的数据, 能看到它的每个特征值都更接近被误分到的那个类而不是本身的类.

从表5和图2可以看出, 与经典决策树ID3算法和C4.5算法相比, 改进后算法具备较好的精度.

4 结 论

对于相当大的范围内的应用来说, 树分类器比其他分类器能生成更精确的分类结果, 特别是先验信息不足的情况下. 但是, 对树分类器的研究相对于神经网络而言要少很多. 一个原因可能是对树分类器的性能的度量和评价更困难. 即使在树分类器的每一等级上的分类效果是最优的, 也不能保证最终的结果是最优的, 因为最终的分类结果与每一级的分类结果并非是线性关系. 所以如果能有效地减少树的层数, 对于分类结果会有很大的提高.

尽管缺少精确的模型来描述决策树, 但是 CART 算法在相当大的程度上给出了构建决策树一个框架. 令 $V = \{(T, F_1, F_2)\}$ 表示一个决策树空间, 其中 T 为训练集, F_1 为分支准则, F_2 为分支停止准则, 则选择不同 F_1 和 F_2 就能够生成不同的决策树. 显然, F_1 和 F_2 与其他的分类方法几乎没有区别. 所以, 即使在决策树中使用神经网络也不足为怪. 基于这种表示, 把其他的分类方法融入到决策树中可能会是一个研究方向.

本文在比较研究了几种经典的决策树算法基础上, 提出了一种改进的决策树算法: 基于度量的决策树(MBDT). 这种决策树实际上是把线性分类器和决策树结合在一起, 能有效地减少决策树的层数, 从而提高决策树的分类效率. 本文通过 MBDT 分类的实验, 验证了上面的结论.

MBDT 需要改进的是: 由于 MBDT 需要搜索整个特征空间, 所以在特征空间很大的时候需要消耗的时间会比较长, 因此, 如何在特征空间上快速的搜索是需要继续研究的问题.

参考文献:

[1] Breiman L, Friedman J h, Olshen R A, et al. Classification and regression trees[M]. California: Wadsworth Publishing Company, 1984.

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

[3] Quinlan J R. C4.5: programs for machine learning[M]. California: Morgan Kaufmann Publishers Inc., 1993.

[4] Freund Y. Boosting a weak learning algorithm by majority [J]. Information and Computation, 1995, 121(2): 256 - 285.

[5] Mehta M, Rissanen J, Agrawal R. MDL-Based decision tree pruning[C]// Int'l Conf. on Knowledge Discovery in Databases and Data Mining (KDD-95). Montreal: [s.n.], 1995.

[6] Mehta M, Agrawal R, Rissanen J. SLIQ: a fast scalable classifier for data mining[C]// Proc. of the Fifth Int'l Conference on Extending Database Technology. France: Avigon, 1996.

[7] Shafer J, Agrawal R, Mehta M. SPRINT: a scalable parallel classifier for data mining[C]// 22nd VLDB Conference. Mumbai, India: Morgan Kaufmann, 1996, 544 - 545.

[8] Breiman L. Bagging predictors [J]. Machine Learning Journal, 1996, 24(2): 123- 140.

[9] Kamber M, Winstone L, Gong W. Generalization and decision tree induction: efficient classification in data mining[C]// Proc. of 1997 Int. Workshop Research Issues on Data Engineering. Birmingham: [s.n.], 1997: 111- 120.

[10] Srivastava A, Singh V, Han E, et al. An efficient, scalable, parallel classifier for data mining [M]. Minneapolis: University of Minnesota, 1997.

[11] 刘小虎, 李生. 决策树的优化算法[J]. 软件学报, 1998, 9(10): 797- 800.

Research and Improvement of Decision Trees Algorithm

FENG Shao-rong*

(School of Computer Science and Engineering, South China University of Technology, Guangzhou 510641, China)

Abstract: Decision tree is a key classification method in data mining. Firstly, based on the research and comparison of several classic decision trees algorithms, an improved decision tree algorithm based on metric is proposed in this paper. In practice, this kind of decision trees combines linear classifier and decision trees. The experimental result indicates that the decision trees based on this method can effectively reduce the decision trees level, which enhance classified efficiency of the decision trees. MBDT classified experiment results demonstrate the correctness and availability of the above conclusions.

Key words: decision trees; metric; ID3 algorithm; entropy