计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(6):198−202 [doi: 10.15888/j.cnki.csa.006954]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

决策树分类算法中 C4.5 算法的研究与改进①

韩存鸽[1,2], 叶球孙[1]

[1](武夷学院 数学与计算机学院, 武夷山 354300)
[2](认知计算与智能信息处理福建省高校重点实验室, 武夷山 354300)
通讯作者: 韩存鸽, E-mail: gezi0401@163.com

摘　要: C4.5 算法是用于生成决策树的一种经典算法, 虽然其有很强的噪声处理能力, 但当属性值缺失率高时, 分类准确率会明显下降, 而且该算法在构建决策树时, 需要多次扫描、排序数据集、以及频繁调用对数, 针对以上缺点, 本文提出一种改进的分类算法. 采用一种基于朴素贝叶斯定理方法, 来处理空缺属性值, 提高分类准确率. 通过优化精简计算公式, 在计算过程中, 改进后的计算公式使用四则混合运算代替原来的对数运算, 减少构建决策树的运行时间. 为了验证该算法的性能, 通过对 UCI 数据库中 5 个数据集进行实验, 实验结果表明, 改进后的算法极大的提高了运行效率.

关键词: 决策树; C4.5 算法; 朴素贝叶斯分类; UCI

引用格式: 韩存鸽,叶球孙.决策树分类算法中 C4.5 算法的研究与改进.计算机系统应用,2019,28(6):198–202. http://www.c-s-a.org.cn/1003-3254/6954.html

Research and Improvement of C4.5 Algorithm in Decision Tree Classification Algorithm

HAN Cun-Ge[1,2], YE Qiu-Sun[1]

[1](School of Mathematics and Computer Science, Wuyi University, Wuyishan 354300, China)
[2](Fujian Provincial Key Laboratory of Cognitive Computing and Intelligent Information Processing, Wuyishan 354300, China)

Abstract: C4.5 algorithm is a classical algorithm used to generate decision tree. Although it has strong noise processing ability, the classification accuracy of C4.5 algorithm decreases obviously when the missing rate of attribute value is high, and the algorithm needs to scan many times when constructing decision tree. This paper presents an improved classification algorithm for sorting data sets and calling logarithms frequently. A method based on naive Bayesian theorem is used to deal with the vacant attribute value and improve the classification accuracy. By optimizing and reducing the calculation formula, the improved formula uses four mixed operations to replace the original logarithmic operation, thus reducing the running time of constructing the decision tree. In order to verify the performance of the algorithm, five data sets in UCI database are tested. The experimental results show that the improved algorithm greatly improves the running efficiency.

Key words: decision tree; C4.5 algorithm; Naive Bayesian classifier; UCI

决策树分类算法是分类挖掘算法中较为常用的算法, 是一种自顶向下的递归算法, 常用于分类器和预测模型. 1966 年 Hunt 等人开发研制的 CLS[1]系统, 为许多决策树算法奠定了基础. 1979 年 Quinlan 提出 ID3[2]算法, 其核心思想以最大的信息增益属性作为分裂属性. 之后的许多决策树算法都是在 ID3 算法的基

① 基金项目: 福建省科技厅自然科学基金 (2017J01406)
Foundation item: Natural Science Foundation of Science and Technology Bureau, Fujian Province (2017J01406)
收稿时间: 2018-12-23; 修改时间: 2019-01-18; 采用时间: 2019-01-22; csa 在线出版时间: 2019-05-25

础上衍生改进的. 其中 C4.5[3] 算法就是在 ID3 算法的基础上发展的, 采用信息增益率作为属性选择的度量依据, 在处理连续性属性的时, C4.5 算法将这些连续属性“离散化”, 解决了 ID3 算法多值偏向性的缺点, 有效避免了取值较多的属性容易被选作分裂属性的趋势, 增强了决策树模型的有效性. 但是当测试集属性值缺失率高时, C4.5 算法的分类准确率会明显下降, 而且在构造树的过程中, 该算法需要多次对数据集进行顺序扫描及排序, 在进行计算时又频繁地调用对数, 增加了算法的时间开销, 从而导致算法低效[4].

文献[5]调整了连续阈值惩罚项, 但是并没有解决决策树过度拟合问题. 文献[6]提出双重熵平均决策树算法,采用熵值拟合的方法, 提高算法运行速度, 但是没有解决空缺属性问题. 文献[7]在连续属性离散化方面进行了改进, 但并没有很好的解决过度拟合问题. 文献[8]适用范围为变精度粗糙集, 不适合一般数据分析.

本文在研究 C4.5 算法的基础上提出一种优化算法, 通过对 UCI 数据库中 5 个数据集进行实验, 实验结果表明, 改进后的算法极大的提高了运行效率.

1 C4.5 算法理论

C4.5 算法是在 ID3 的基础上加进了对连续型属性、属性值空缺情况的处理, 通过生成树和树剪枝两个阶段来建立决策树. C4.5 算法通过计算每个属性的信息增益率 (information GainRatio), 最后选出具有最高信息增益率的属性给定集合 S 的测试属性, 再根据测试属性值建立分支, 采用递归算法, 得到初步决策树. C4.5 算法的相关计算公式如下[9]:

(1) 期望信息 (也称信息熵) 设 S 是 S_i 个数据样本的集合, 假定类标号属性有 m 个不同值, 定义 m 个不同类 T_i (i=1, ⋯, m). 设 S_i 是类 T_i 中的样本数. 对一个给定的样本分类所需的期望值为:

$$Info(S) = -\sum_{i=1}^{m} \frac{s_i}{s} \log_2 \frac{S_i}{S} \tag{1}$$

(2) 信息增益 由属性 A 划分成子集的信息量为:

$$E(A) = Info_A(S) = \sum_{j=1}^{v} \frac{S_i}{S} Info(S_{ij}) \tag{2}$$

信息增益为原来的信息需求与新的需求之间的差.

$$即 Gain(A) = Info(S) - E(A) \tag{3}$$

(3) 信息增益率 C4.5 算法中引入了信息增益率, 属性 A 的信息增益率的计算公式为:

$$GainRatio(A) = \frac{Gain(A)}{SplitE(A)} \tag{4}$$

$$SpliE(A) = -\sum_{i=1}^{k} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|} \tag{5}$$

C4.5 算法采用后剪枝技术对生成的决策树进行剪枝操作, 形成决策树模型, 根据建立好的模型, 生成一系列 IF-THEN 规则, 实现对训练集的分类.

2 改进的 C4.5 算法

2.1 决策分类过程中空缺属性值处理研究

(1) 延伸贝叶斯分类算法

决策树进行分类时, 测试样本数据中可能存在缺失某些属性值. 传统的决策树算法在处理缺失属性值时, 一般采用抛弃缺失属性值的样本, 或重新赋予训练样本该属性的常见值[10]. C4.5 算法采用概率分布的填充法来处理缺失属性值, 具体执行过程: 首先为某个未知属性的每个可能值赋予一个概率, 根据某节点上属性不同值的出现频率, 这些概率可以再次估计[11]. 虽然 C4.5 算法有很强的处理噪声数据的能力, 但当训练集合属性值的缺失率很高时, 使用该算法构建的决策树模型的结点数更多, 模型更为复杂, 而且分类准确率也会下降[12]. 鉴于朴素贝叶斯分类具有坚实的理论基础、较小的出错率. 本文提出一种基于朴素贝叶斯定理方法[13], 来处理空缺属性值.

假设数据样本为 n 维向量空间, V={V_1, V_2, V_3, ⋯, V_i} 若该数据样本有 C_1, C_2, C_3, ⋯, C_m 个类别属性取值, 那么基本贝叶斯将未知类别的样本 V 归到类别 C_i, 当且仅当:

$$\frac{p(c_i)p(v|c_i)}{p(v)} > \frac{p(c_j)p(v|c_j)}{p(v)}$$

等价$p(c_i)p(v|c_i) > p(c_j)p(v|c_j)$.

若 V 的 m 个属性互相独立, 那么

$$p(v|c_i) = \prod_{k=1}^{n} p(v_k|c_i)$$

继而推出:

$$p(c_i)\prod_{k=1}^{n} p(v_k|c_i) > p(c_j)\prod_{k=1}^{n} p(v_k|c_j)$$

所以, $\frac{s_i}{s}\prod_{k=1}^{n} p(v_k|c_i) > \frac{s_j}{s}\prod_{k=1}^{n} p(v_k|c_j)$(因为$p(c_i) = \frac{s_i}{s}$).

故将未知类别的样本 X 归到类别 Ci,当且仅当

$$\frac{s_i}{s}\prod_{k=1}^{n}p(v_k|c_i) > \frac{s_j}{s}\prod_{k=1}^{n}p(v_k|c_j),\ 1<=j<=m \text{ 且 } j\neq i$$

根据上述原理可以实现对多维空缺属性的处理:

$$\frac{s_i'}{s}\prod_{k=1}^{n}p(v''_k|c'_i) > \frac{s'_j}{s}\prod_{k=1}^{n}p(v''_k|c'_j)$$
$$1<=j<=m \text{ 且 } j\neq i$$

(2) 空缺属性值处理过程

本文采用上述算法处理空缺属性, 大致流程如下:

① 读入测试数据.

② 若数据集中没有空缺属性值, 则把数据压入 D_i 集合; 如果测试数据集中含有空缺属性值, 采用“样本空缺属性个数排序, 越少越排前, 反之则排后. ”的原则插入到 D_2 集合.

③ 返回第一步, 直到测试数据集中所有数据都检测结束. 最后形成 D_i、D_2 两个集合. 其中 D_i 中存放没有空缺属性值, D_2 中存放包含空缺属性的信息.

④ 取出 D_2 中的记录数据, 为缺失属性赋值计入 D_i.

⑤ 递归调用上述第④步, 直到 D_2 中记录为 0.

2.2 改进思想及改进计算方式

虽然 C4.5 算法具有建模速度快、准确率高、易于理解、灵活简单等优点, 但是该算法在构造树的过程中, 需要多次对数据集进行顺序扫描及排序,在进行计算时又频繁地调用对数, 增加了算法的时间开销, 从而导致算法低效. 针对以上缺点, 本文对 C4.5 算法的计算公式进行精简优化, 从减小计算时间上提出一种改进算法命名为 N-C4.5 算法. 公式简化如下:

假设 S 是 n 维有穷离散数据集, S 中只有正反两种情形, 分别表示为 YS 、NS, 其中正例 YS 的记录数为 m, 反例 NS 的记录数为 n, 根据式 (1) 数据集 S 分类所需的期望值为:

$$Info(S) = -\frac{m}{m+n}\log_2^{\frac{m}{m+n}} - \frac{n}{m+n}\log_2^{\frac{n}{m+n}}$$

假设决策树的根为 A 属性, A 属性的取值为$\{A_1, A_2, \cdots, A_i\}$, 利用属性 A 将数据集 S 分为$\{S_1,S_2, \cdots, S_i\}$个子集, 其中 S_i 为 S 中属性 A 取 A_i 值的样本实例数据, 那么由属性 A 划分成子集的信息量为

$$I(S_i) = -\frac{m_i}{m_i+n_i}\log_2^{\frac{m_i}{m_i+n_i}} - \frac{n_i}{m_i+n_i}\log_2^{\frac{n_i}{m_i+n_i}}$$

在数据集 D 中计算属性 X 的信息熵公式为

$$Info_A(S) = \sum_{i=1}^{v}\frac{m_i+n_i}{m+n}I(S_i)$$
$$= \frac{1}{(m+n)\ln 2}\sum_{i=1}^{v}-m_i\ln\frac{m_i}{m_i+n_i} - n_i\ln\frac{n_i}{m_i+n_i}$$

$$Info_A(S) = \sum_{i=1}^{v}\frac{m_i+n_i}{m+n}I(S_i)$$
$$= \frac{1}{(m+n)\ln 2}\sum_{i=1}^{v}(-m_i\ln\frac{m_i}{m_i+n_i} - n_i\ln\frac{n_i}{m_i+n_i})$$

$$Info_A(S) = \sum_{i=1}^{v}(-m_i\ln\frac{m_i}{m_i+n_i} - n_i\ln\frac{n_i}{m_i+n_i})$$

(由于$\frac{1}{(m+n)\ln 2}$为常数, 故可以简化).

根据无穷小原理$\ln(1+x)\approx x$, 那么:

$$\ln\frac{m_i}{m_i+n_i} = \ln(1-\frac{n_i}{m_i+n_i}) = -\frac{n_i}{m_i+n_i}$$

同理,$\ln\frac{n_i}{m_i+n_i} == -\frac{m_i}{m_i+n_i}$.

$$Info_A(S) = \sum_{i=1}^{v}\left(-m_i\ln\frac{m_i}{m_i+n_i} - n_i\ln\frac{n_i}{m_i+n_i}\right)$$
$$= \sum_{i=1}^{v}\left(\frac{m_in_i}{m_i+n_i} + \frac{m_in_i}{m_i+n_i}\right) = \sum_{i=1}^{v}\frac{2m_in_i}{m_i+n_i}$$

$$Info_A(S) = \sum_{i=1}^{v}\frac{m_in_i}{m_i+n_i}(\text{常数2省略})$$

信息增益: $Gain(A) = Info(S) - Info_A(S)$.

$$SplitE(A) = -\frac{n_i}{m_i+n_i}\log_2^{\frac{n_i}{m_i+n_i}} - \frac{m_i}{m_i+n_i}\log_2^{\frac{m_i}{m_i+n_i}}$$
$$= \sum_{i=1}^{v}\frac{2m_in_i}{\ln 2(m_i+n_i)^2}$$

$$SplitE(A) = \sum_{i=1}^{v}\frac{m_in_i}{(m_i+n_i)^2}\left(\text{因为}\frac{2}{\ln 2}\text{为常数可简化}\right)$$

信息增益率: $GainRatio(A) = \frac{Info(S)-Info_A(S)}{SplitE(A)}$.

改进后的计算公式使用四则混合运算代替原来的对数运算, 可以使计算效率得到提升. 原 C4.5 算法的时间复杂度为 $O(n^2\log_2^n)$, 改进后的算法时间复杂度为 $O(n^2)$, 新算法去掉了对数运算, 优化了时间复杂度.

2.3 N-C4.5 算法实际应用

为了验证算法正确性, 这里以顾客购买计算机的样本信息 S 为例, 具体样本记录如表 1 所示. 使用改进

后的算法构建决策树, 命名为 N-C4.5 算法. 该数据集 S 有 age、income、student、credit_rating、buy 5 个属性, 其中 buy 为类标号属性, 有两个取值{yes,no}, S 数据集的前 4 个属性为分裂属性, 并且每个属性都是离散化的. 这里根据每个人的 age、income、student、credit_rating 来进行分类, 判断顾客是否购买计算机.

表 1 顾客购买计算机的样本信息 S

RID	age	income	student	credit_rating	buy
1	youth	high	no	bad	no
2	youth	high	no	good	no
3	middle	high	no	bad	yes
4	senior	medium	no	bad	yes
5	senior	low	yes	bad	yes
6	senior	low	yes	good	no
7	middle	low	yes	good	yes
8	youth	medium	no	bad	no
9	youth	low	yes	bad	yes
10	senior	medium	yes	bad	yes
11	youth	medium	yes	good	yes
12	middle	medium	no	good	yes
13	middle	high	yes	bad	yes
14	senior	medium	no	good	no

对客户购买计算机的样本信息按照改进以后的算法构建决策树, 利用改进后的公式计算信息熵、信息增益、分裂信息量、以及信息增益率, 同时选择最大信息增益率作为分裂结点, 采用递归算法构建决策树. 在数据集 S 中属于类别“yes”的有 9 个, 属于类别“no”的有 5 个, 根据改进后的公式计算过程分如下五步.

Step1. 计算分类所需的期望信息

$$Info(S)=\frac{mn}{(m+n)^2}=\frac{9\times5}{(9+5)^2}=0.230$$

Step2. 计算每个属性的信息熵

$$Info(age)=\frac{2\times3}{2+3}+\frac{4\times0}{4+0}+\frac{3\times2}{3+2}=2.4$$
$$Info(income)=3.083$$
$$Info(student)=2.571$$
$$Info(credit\_rating)=3.0$$

Step3. 计算每个属性的信息增益

$$Gain(age)=Info(S)-Info(age)=0.230-2.4=-2.170$$
$$Gain(income)=0.230-3.083=-2.783$$
$$Gain(student)=0.230-2.571=-2.341$$
$$Gain(credit\_rating)=0.230-3.0=-2.8$$

Step4. 计算每个属性的分裂信息度量

$$SplitE(age)=\frac{5\times4\times5}{(5+4+5)^2}=0.510$$
$$SplitE(income)=0.490$$
$$SplitE(student)=0.250$$
$$SplitE(credit\_rating)=0.245$$

Step5. 信息增益率

$$GainRatio(age)=\frac{-2.170}{0.510}=-4.255$$
$$GainRatio(income)=-5.680$$
$$GainRatio(student)=-9.346$$
$$GainRatio(credit\_rating)=-11.430$$

由计算结果可知, age 属性的信息增益率最大, 所以选择 age 作为分裂属性, 将原数据集分为 youth、middle、senior 三个子集, 采用递归算法计算每个属性的信息熵、信息增益、分裂信息量、信息增益率, 对于 youth 子集, 经计算得出 student 信息增益率最大, 选择 student 作为测试属性; 对于 middle 子集全部归为 yes 类, 所以不需要进行分类; 而对于 senior 子集, 通过计算得知, credit_rating 信息增益率最大, 因此选择 credit_rating 作为测试属性, 根据所有结点, 可以得出购买计算机数据集采用 N-C4.5 算法构造的决策树如图 1.

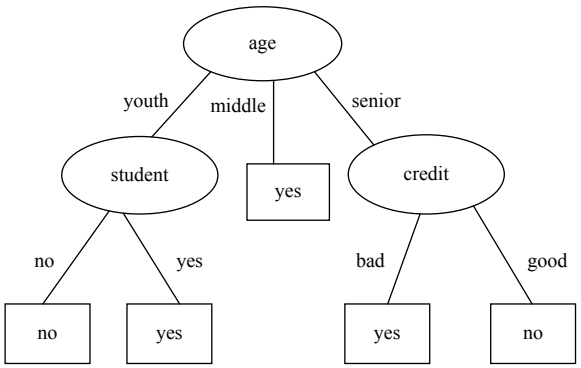

图 1 N-C4.5 算法构造的决策树

由图 1 可以看出采用 N-C4.5 算法构造的决策树与使用原 C4.5 算法得到的决策树结构完全相同, 但 C4.5 算在计算的过程中调用了大量的对数, 而 N-C4.5 算法却只需要采用简单的混合运算即可, 因此运算效率大幅度提高.

3 实验结果比较分析

为了验证 N-C4.5 算法的性能, 本实验选择 UCI 数据库中 5 个数据集进行仿真实验, 实验数据的基本信息见表 2 , 实验硬件配置: 内存 4 GB, CPU 2.50 GHz , 使用 Window7 操作系统, 采用 Java 语言在 WEKA 平

台中进行仿真实验. 仿真实验时, 首先利用延伸的贝叶斯分类算法填充多维空缺属性, 该方法会生成空缺属性的各种可能取值组合, 接着遍历各种组合, 按公式计算值并记录最大值即可; 然后利用改进后的公式计算信息熵、信息增益、分裂信息量、以及信息增益率, 最后选择最大信息增益率作为分裂结点, 采用递归算法构建决策树.

表 2 测试数据集基本信息

Dataset	Example	Missing	Featrues	classes
Iris	150	Sepalwidth(30)	4	3
Glass	214	Silicon(45)	10	6
Vote	435	Crime(50)	16	2
Diagnostic	569	Area(45)	32	2
Breast	699	Ino-nodes(75)	9	2

Iris 数据集是构建决策树挖掘中最典型的一组数据, 共有 150 条记录, 有 4 个分裂属性, 类标记属性为 3 种, 人为添加部分缺失信息, 缺失属性为 sepal width, 共有 30 条缺失记录, 使用 Iris 数据集对改进前后的算法进行 5 次仿真实验, 实验结果如图 2 所示.

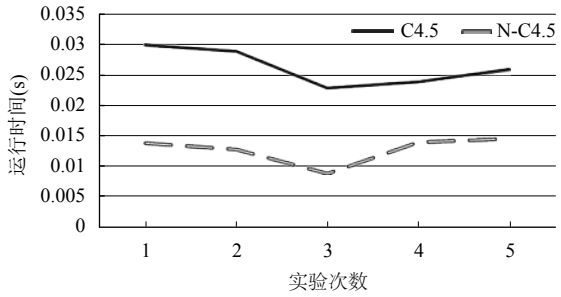

图 2 Iris 数据建模时间

由图 2 可以看出, 对于 Iris 数据, N-C4.5 算法的运行时间短, 优化效果明显. 为了验证 N-C4.5 算法算法的效率, 对于其他 4 组数据集, 同样都分别使用改进后的 N-C4.5 算法与 C4.5 算法进行仿真实验, 测试时每个数据集都运行 5 次, 建模时间取 5 次的平均值, 建模时间如表 3 所示.

表 3 建模信息表

Dataset	C4.5	N-C4.5	比率	提升 (%)
Iris	0.026	0.012	2.17	0.3
Glass	0.85	0.71	1.12	0.1
Vote	0.018	0.012	1.5	0.15
Diagnostic	0.95	0.78	1.22	0.1
Breast	3.43	1.62	2.12	0.03

由表中可以看出, 数据样本的大小会影响算法的运行时间, 一般情况下, 测试数据样本数据量越大, 需要运行的时间越长. 算法改进前后的比率都大于 1, 正确率的提升都是正值, 可以得出结论: N-C4.5 算法的建模时间明显优于 C4.5 算法, 分类准确率也高于原 C4.5 算法.

4 总结

虽然 C4.5 算法具有建模速度快、准确率高、易于理解、灵活简单、以及处理噪声数据的能力强等优点, 但当属性值缺失率高时, 分类准确率会明显下降; 在构建决策树时, 需要多次扫描、排序数据集、以及频繁调用对数等缺点. 本文提出一种改进的决策树分类算法. 采用一种基于朴素贝叶斯定理方法, 来处理空缺属性值, 提高分类准确率. 通过优化计算公式, 改进后的 N-C4.5 算法使用四则混合运算代替原来的对数运算, 减少构建决策树模型的运行时间, 使计算效率得到提升.

参考文献

1 芦思雨. 数据挖掘中分类算法的比较分析[硕士学位论文]. 天津: 天津财经大学, 2016.
2 唐一. ID3 改进算法在高校就业系统中的应用研究[硕士学位论文]. 长沙: 湖南大学, 2012.
3 陈杰, 邬春学. 决策树 C4.5 算法改进与应用. 软件导刊, 2018, (10): 88–92.
4 黄秀霞. C4.5 决策树算法优化及其应用[硕士学位论文]. 无锡: 江南大学, 2017.
5 徐鹏, 林森. 基于 C4.5 决策树的流量分类方法. 软件学报, 2009, 20(10): 2692–2704.
6 蔡星. 决策树算法及其改进. 科技创新导报, 2014, (12): 40, 45.
7 姚亚夫, 邢留涛. 决策树 C4.5 连续属性分割阈值算法改进及其应用. 中南大学学报 (自然科学版), 2011, 42(12): 3772–3776.
8 刘兴文, 王典洪, 陈分雄. 一种基于变精度粗糙集的 C4.5 决策树改进算法宰. 计算机应用研究, 2011, 28(10): 3649–3651. [doi: 10.3969/j.issn.1001-3695.2011.10.011]
9 Han JW, Kamber M. 数据挖掘: 概念与技术. 范明, 孟小峰, 译. 北京: 机械工业出版社, 2007.
10 乐明明. 数据挖掘分类算法的研究和应用[硕士学位论文]. 成都: 电子科技大学, 2017.
11 李旭. 五种决策树算法的比较研究[硕士学位论文]. 大连: 大连理工大学, 2011.
12 缪连芬. 改进的 C4.5 算法在大学生情感素质分析中的研究与应用[硕士学位论文]. 上海: 上海师范大学, 2018.
13 李锦善. 基于 Volume Test 的贝叶斯分类器研究[硕士学位论文]. 北京: 北京交通大学, 2008.