
北京航空航天大學
BEIHANG UNIVERSITY

# 模式识别与机器学习大作业
# 决策树


院(系)名称　　高等理工学院

学　　号　　16231235

姓　　名　　李谨杰

指 导 教 师　　秦曾昌


2019 年 6 月

# 摘要

本文首先介绍了信息熵、信息增益的概念，又引出 ID3 决策树的基本原理。之后编写 MATLAB 程序对决策树的性质进行探究，并详细解释程序流程。接着，通过控制每个分支节点允许的最小数据量，改变决策树的大小，探究决策树大小与识别正确率的关系。最终得出结论：决策树的大小与识别正确率没有必然联系，但几乎所有的经典决策树都可以在正确率不增加的前提下缩减树的大小，提升泛化能力，由此我深入理解了剪枝的概念。

文章中还介绍了基尼系数、增益比等纯度判定条件，列举了常用的剪枝方法，同时对比了决策树与朴素贝叶斯方法，说明了决策树模型的优点和缺点。

**关键字：** 决策树、信息增益

# 目录

# 一、引言

决策树归纳法是最简单、最成功的学习算法之一。决策树由内部和外部节点组成，节点之间的互连称为树的分支。内部节点是一个决策单元，根据相关变量的不同可能值来决定下一次应该访问哪个子节点。相反，外部节点也被称为叶节点，是分支的终止节点。它没有子节点，并且与描述给定数据的标签相关联。决策树是树结构中的一组规则，其中的每个分支都可以解释为与沿着该分支访问的节点关联的决策规则。

# 二、决策树原理讲解

## 2.1 概述

决策树通过从根节点到叶节点对实例进行分类，这种树结构的分类器将数据集的输入空间递归地划分为互斥空间。按照这种结构，每个训练数据都被标识为属于某个子空间，该子空间被分配一个标签、一个值或一个动作来描述其数据点。决策树机制具有很好的透明度，因为我们可以很容易地遵循树结构来解释决策是如何做出的，因此，当我们明确描述树的条件规则时，可以增强解释能力。

随机变量的熵是通过观察其值产生的平均信息量。考虑抛硬币的随机实验，头的概率等于 0.9，所以 p（正）=0.9，p（反）=0.1。这比 P（正）=0.5 和 P（正）=0.5 的情况提供了更多的信息。

熵用于评价物理中的随机性，大的熵值表示这个过程是非常随机的。根据每个属性的信息内容，对决策树进行启发式引导，我们采用信息熵评估每个属性的信息，这种评估可以作为分类的一种手段。假设我们有 m 类，对于一个特定的属性，我们用 pi 表示它，用属于 ci 类的数据的比例表示，其中 i=1，2，…m。这个分布的熵被定义为

$$\text{Entropy} = \sum_{i=1}^{m} -p_i \cdot \log_2 p_i$$

选择以 2 为底，是因为计算机以二进制进行计算。我们也可以说，信息熵是对一组训练数据集中杂质的度量：信息熵越大，数据越不纯净。基于信息熵，信息增益（IG）被用来衡量用某种属性进行类划分的有效性。

$$IG(S,A) = \text{Entropy}(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \text{Entropy}\left(S_v\right)$$

信息增益越大，表示分割之后的数据越纯净，分割越彻底。决策树实际上是在高维空间中对数据进行划分，可视化效果如图 2.1.

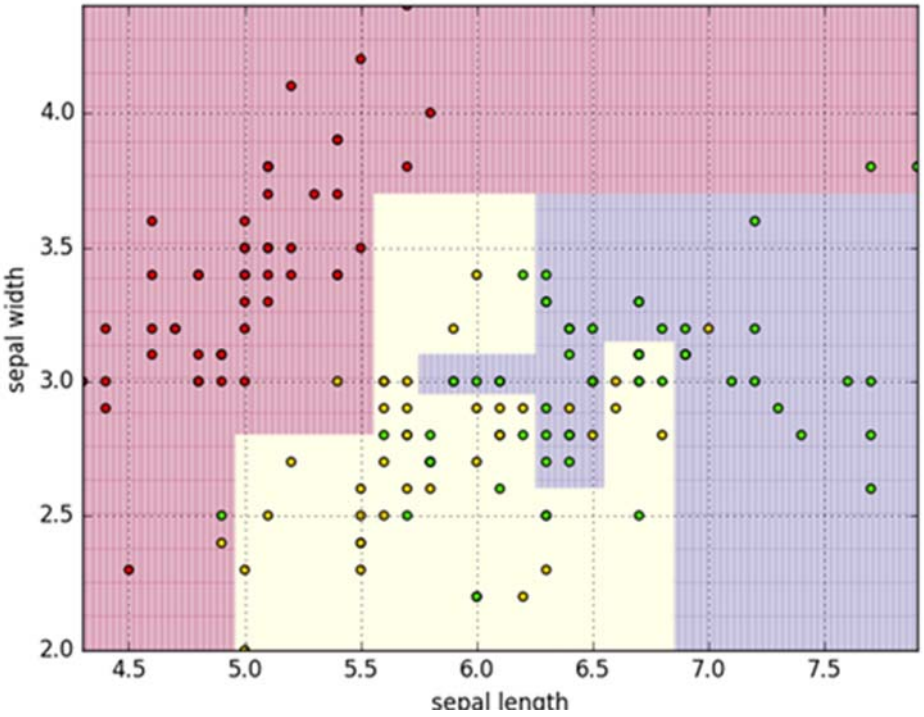


图 2.1 决策树的可视化

其中红色区域为花 1，黄色区域为花 2，蓝色区域为花 3。决策树即为在四维空间中构造分割这些区域的超平面。

2.2 其他类型决策树的介绍

如何选取最优分割，决定了决策树的种类。本次实验采用信息增益作为评价标准，属于 ID3 决策树算法。另外，为减少对可取值数目较多的属性有所偏好的影响，[Quinlan,1993]提出了使用“增益率”进行分割的算法，即 C4.5 决策树算法。除此以外，也有采用基尼系数进行判断的 CART 决策树算法。

2.3 决策树的剪枝

为避免决策树的过拟合，提高泛化性能，需要对决策树进行“剪枝”处理。剪枝策略分为两种：“预剪枝”和“后剪枝”。

预剪枝：在构造决策树的过程中，先对每个结点在划分前进行估计，如果当前结点的划分不能带来决策树模型泛化性能的提升，则不对当前结点进行划分并且将当前结点标记为叶结点。

后剪枝：先把整棵决策树构造完毕，然后自底向上对非叶结点进行考察。若将该结点对应的子树换为叶结点能够带来泛化性能的提升，则把该子树替换为叶结点。

在本实验中，由于训练数据本身数量较少，我采用控制每一类别最少个数的方法控制决策树的大小，属于预剪枝，以探究决策树的大小对判断正确率的影响。

2.4 Iris 数据集简介

Iris 数据集是常用的分类实验数据集，由 Fisher, 1936 收集整理。Iris 也称鸢尾花卉数据集，是一类多重变量分析的数据集。数据集包含 150 个数据集，分为 3 类，每类 50 个数据，每个数据包含 4 个属性。可通过花萼长度，花萼宽度，花瓣长度，花瓣宽度 4 个属性预测鸢尾花卉属于（Setosa，Versicolour，Virginica）三个种类中的哪一类。

## 三、决策树算法详解

定义函数 train_tree，节点以结构体形式进行保存。传入训练数据，父节点标号，节点结构体。

a) 计算当前数据的总信息熵 E(s)，统计标签集合为 A

b) 如果 E(s)为零，或样本的总数小于设定的值，则将该节点生成为叶节点，其类别标记为标签集合 A 中数量最多的类别。将标号传回。

c) 如果不满足上述叶节点条件，则

For 数据的每一个属性

For 数据的全部可能划分（从最小划分以 0.1 为步长直到最大划分）

根据当前的划分得到两组数据集

根据这些数据集计算信息增益 IG

End

End

d) 选取信息增益最大的划分。本数据集只有三个标签，如果按二分法划分时，必有一个子节点具有某种类别。由于数据离散，如果出现多个划分的信息增益相同时，选择使单类别节点的取值范围尽可能大的那种划分。

e) 按最优划分将训练数据分成 a、b 两组，递归训练 a。

f) 当 a 全部训练完毕，传回最新生成的节点标号 i，继续递归训练 b，子节点标号从 i+1 开始。

**程序注意事项：**

（1） 每次计算信息增益结束，要清空新生成的两组数据。否则会出现数据之间的覆盖，影响结果。

（2） 当多组划分的信息增益相同时，此时对划分的选择会影响最后的检测结果。经过我的测试，选择靠后的那组划分效果是最好的。

（3） 一定要注意节点标号。我的策略是，将其中一个分支的子节点全部展完，再生成另外一个分支，顺序标号。

## 四、实验结果

将实验数据均分成两组，一组用于训练，另一组用于测试。因为原样本是按 50setosa、50versicolor、50virginica 的顺序排列的，所以需要随机生成训练和测试数据集。编写 MATLAB 代码，不考虑剪枝，生成决策树如下：

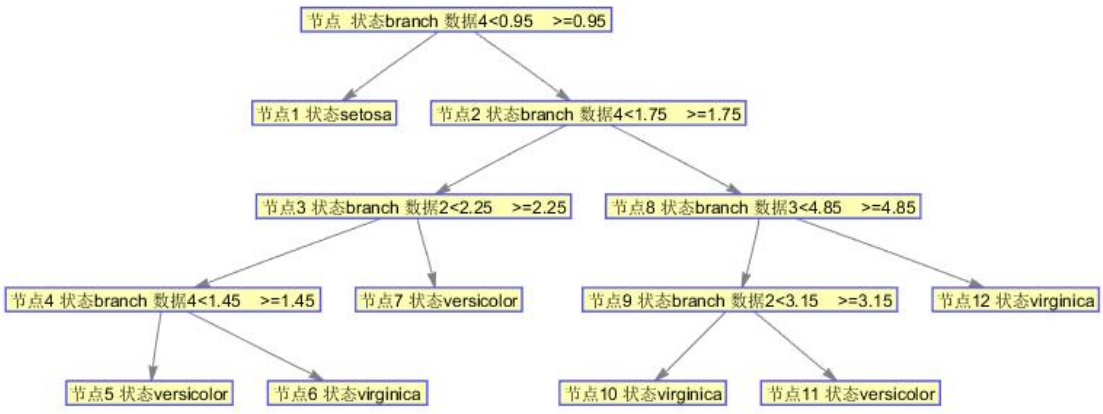


图 4.1 无剪枝决策树

正确率 93.33%。

使用 MATLAB 自带程序生成决策树如下：

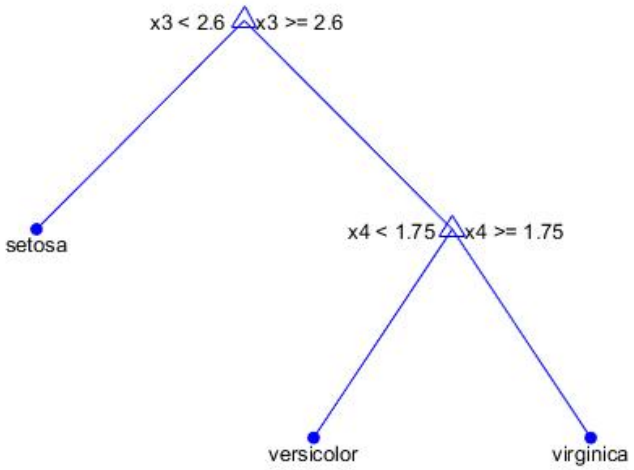


图 4.2 matlab 自带函数生成的决策树

正确率 94.67%。

从这两张图的对比可以看出，经过剪枝处理的简单决策树反而比图 4.1 中的经典决策树正确率要高。这是因为划分非常彻底的决策树会错误包含噪声，造成过拟合现象，降低正确率。

改变每个分支节点所含的最小数据量，得到决策树的分类正确率如下：

| 最小数据量 | 0 | 10 | 20 | 30 | 40 |
|---|---|---|---|---|---|
| 节点总数 | 13 | 9 | 9 | 5 | 5 |
| 叶节点总数 | 7 | 5 | 5 | 3 | 3 |
| 正确率 | 93.33% | 93.33% | 93.33% | 94.67% | 94.67% |

当分支节点的最小数据量为 30 时，此时的决策树如图 4.3：

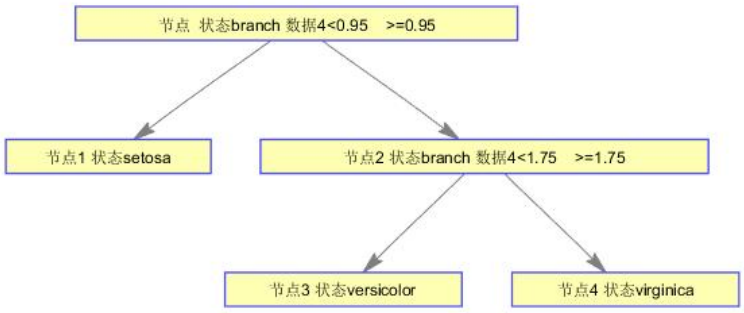


图 4.3 分支节点最小数据量为 30 时的决策树

对比图 4.3 与图 4.2，可以看到剪枝后的决策树与 MATLAB 自带函数生成的决策树非常相近，正确率也提高到相同的程度。

改变训练数据与测试数据，得到决策树分支节点所含最小数据量与正确率的关系如下表：

实验次数 2

| 最小数据量 | 0 | 10 | 20 | 30 | 40 |
|---|---|---|---|---|---|
| 节点总数 | 15 | 9 | 9 | 5 | 5 |

| | | | | | |
|---|---|---|---|---|---|
| 叶节点总数 | 8 | 5 | 5 | 3 | 3 |
| 正确率 | 98.67% | 98.67% | 98.67% | 96.00% | 96.00% |

实验次数 3

| | | | | | |
|---|---|---|---|---|---|
| 最小数据量 | 0 | 10 | 20 | 30 | 40 |
| 节点总数 | 11 | 9 | 9 | 5 | 5 |
| 叶节点总数 | 6 | 5 | 5 | 3 | 3 |
| 正确率 | 90.67% | 90.67% | 90.67% | 93.33% | 93.33% |

实验次数 4

| | | | | | |
|---|---|---|---|---|---|
| 最小数据量 | 0 | 10 | 20 | 30 | 40 |
| 节点总数 | 9 | 7 | 7 | 5 | 5 |
| 叶节点总数 | 5 | 4 | 4 | 3 | 3 |
| 正确率 | 97.33% | 97.33% | 97.33% | 92.00% | 92.00% |

从这些实验结果我们可以得到以下结论：

（1） 树的大小与判断正确率之间没有必然联系。对于有的数据，缩小树的规模会提升分类正确率；但对于另外一些数据，缩小树的规模反而会降低分类正确率。尽可能增多训练数据可以得到更精准的决策树模型。

（2） 从结果上判断，当限制分支节点的最小数据量从 0 变为 20 时，所有树的规模都有一定程度的缩小，同时不改变判断正确率。根据奥卡姆剃刀原则，在相同的正确率的前提下，简单的树比复杂的树要好。因此，对决策树进行剪枝十分必要。

（3） 总结实验结果，剪枝的方法就是在不减少正确率的前提下，将一部分对提高正确率没有作用的分支节点改为叶节点。

## 五、对结果的讨论

Stage1：我使用结构体表示每个节点，每个结构体的序号即为这个节点的标号。结构体中存有父节点和子节点的标号。以此数据结构来实现决策树。

Stage2：

（1）对于连续特征，二分处理将其离散化。即选取一个门限值，小于这个值的数据算作第一类，大于这个值的数据算作第二类。如果要分成多个类别，也可以选取多个门限值将连续区间分割，从而将连续分布离散化。离散属性使用一次后无法重复使用，但连续属性还可作为其后代节点的划分属性。

（2）关于树的大小与识别准确率的观点呈现在第五章-实验结果中。

（3）对于此问题，我理解为：当原始数据集已经经过一定分类后，剩余数据集中某些类型的数据可能只有很少的几个，这些数据可能是噪声造成的，但是传统的决策树会将这些异常数据生硬的分到新的类别中，从而降低泛化性能。对于这种问题，我觉得只有“剪枝”的方法可以解决。一般文献中“剪枝”的方法如第四节结论（3）中所写，确保正确率不下降。在本次实验中，我采用的方法是确保分支节点的数据量均在一个范围之上，避免噪声的干扰。除此之外，还可以控制数据集的信息熵大于某个数值，这也是控制决策树大小的一种方法。这几种方法均能提升决策树的鲁棒性。

（4）决策树与朴素贝叶斯方法的对比：

朴素贝叶斯分类法：

对于给定的训练数据集，首先基于特征条件独立假设，估计类先验概率 P(c)，并为每个属性估计条件概率$P(x_i|c)$；然后基于此模型，对给定的输入 x，利用贝

叶斯定理，求出后验概率最大的输出 y,即 $h(\mathbf{x}) = \underset{c \in y}{\operatorname{argmax}} P(c) \prod_{i=1}^{d} P(x_i \mid c)$ 。

优点：与朴素贝叶斯分类法相比，不依赖属性条件独立性假设，适用的数据范围广，即使属性间有联系也适用；容易构造规则，且这些规则易于解释和理解；可以扩展到大型数据库中。

缺点：决策树是基于信息熵的概念而提出的，数学背景不如朴素贝叶斯模型稳固；需要估计的参数较多，且参数和数据的分布、剪枝规则有关，分类效果不稳定；存在过度拟合问题。

## 六、收获、体会及建议

经过本次实验，我深入理解了决策树的原理，将课上讲的知识学以致用，锻炼了编程能力，收获很大。机器学习的算法有很多，只有付诸实践才能真正领会这些算法的精神。

## 七、参考文献

[1] 机器学习课件 秦曾昌
[2] 周志华. 机器学习. 清华大学出版社
[3] 百度百科：IRIS （IRIS 数据集）https://baike.baidu.com/item/IRIS/4061453
[4] 实验指导书：Guides to a Series of Experiments