

课程号: 180206081100M1001H-01

第16章

决策树方法

Classification Methods with Decision Trees

向 世 明

smxiang@nlpr.ia.ac.cn

<https://people.ucas.ac.cn/~xiangshiming>

时空数据分析与学习课题组 (STDAL)

中科院自动化研究所 多模态人工智能系统国家重点实验室

助教: 杨 奇 (yangqi2021@ia.ac.cn)

张 涛 (zhangtao2021@ia.ac.cn)

Top 10 algorithms in data mining

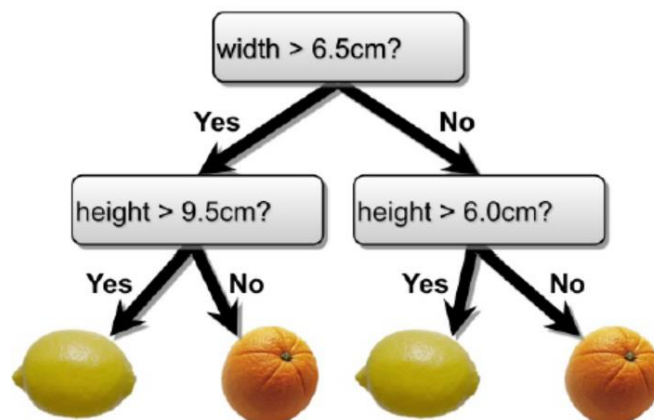
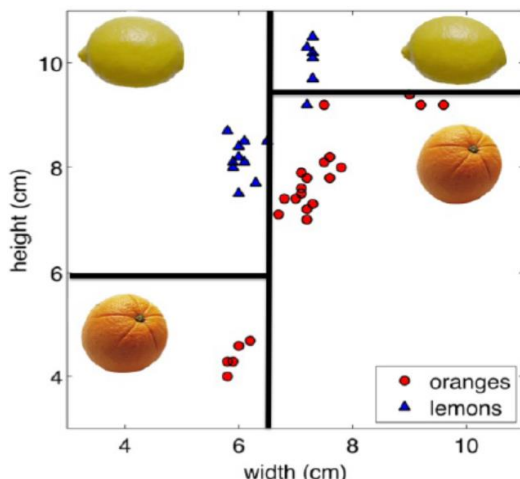
IEEE Int'l Conf. on Data Mining (ICDM), 2006, Hong Kong:

1. **C4.5**: decision trees classifiers
2. **K-Means**: simplest clustering algorithm
3. **SVM**: support vector machine for classification
4. **Apriori**: a seminal algorithm for finding frequent item-sets (关联规则挖掘算法)
5. **EM**: maximum likelihood estimation of mixture distribution
6. **PageRank**: search ranking algorithm using hyperlinks (Google)
7. **AdaBoost**: one of the most important ensemble methods
8. **KNN**: simplest classifier
9. **Naïve Bayes**: simplest Bayes, independent Bayes
10. **CART**: classification and regression Trees

16.1 决策树的基本概念

16.1.1 决策树

- 分类模型：
 - 常用方法：Bayes decision rule, SVM, nearest neighbors ... **Any other idea?**
- **Decision Tree (How to construct):**
 - **Pick an attribute, do a simple test (选择一个属性, 做分类测试)**
 - Conditioned on a choice, pick another attribute, do another test (基于当前选择, 再选择一个属性, 再做测试) (**递归选择最优化属性**)
 - Assign a class each leaf
 - Do other branches as well

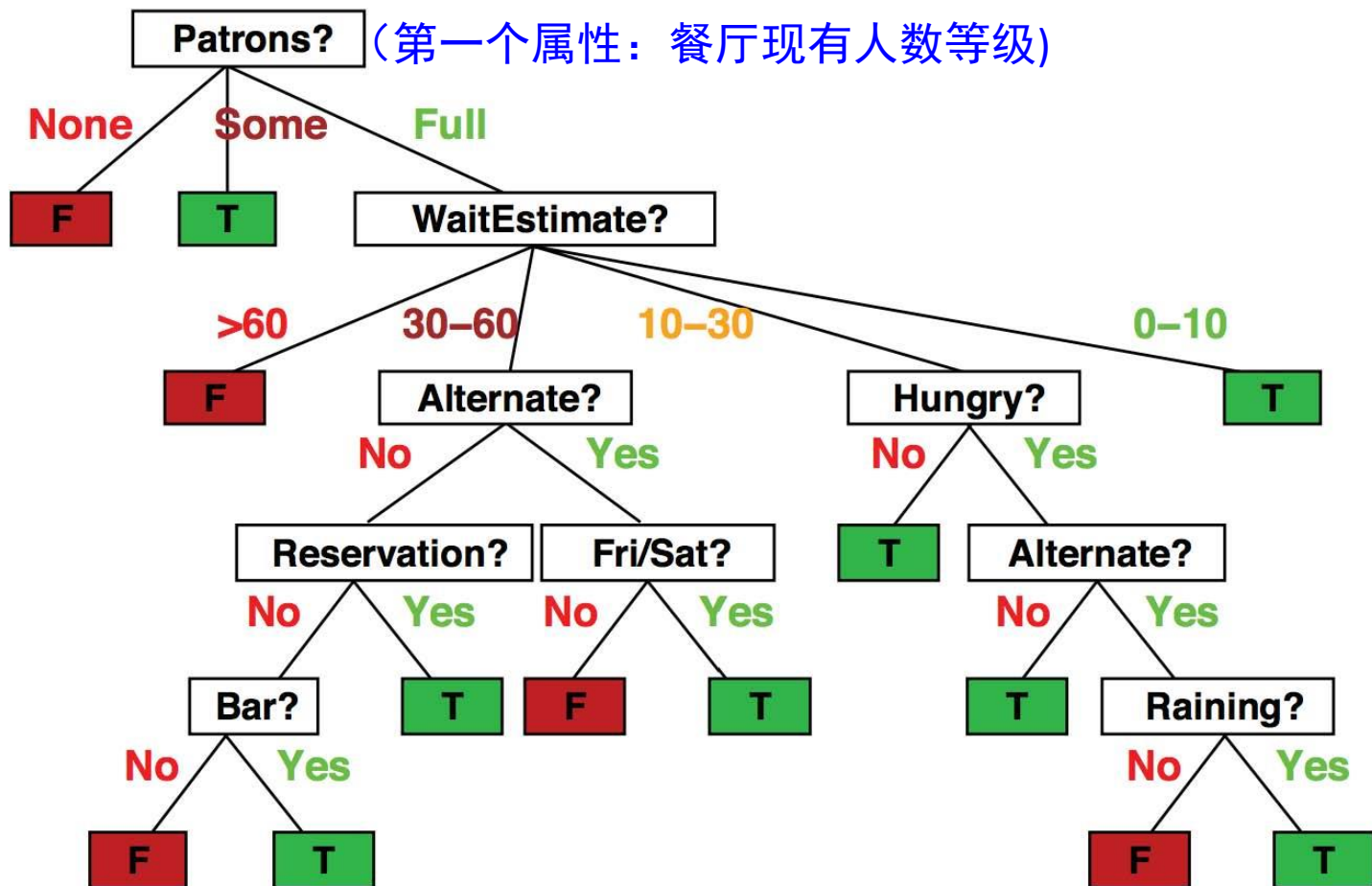


Example 1: with Discrete Inputs （等餐问题）

Example	Input Attributes (10个属性)										Goal
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>WillWait</i>
x_1	Yes	No	No	Yes	Some	\$\$\$	No	Yes	French	0-10	$y_1 = \text{Yes}$
x_2	Yes	No	No	Yes	Full	\$	No	No	Thai	30-60	$y_2 = \text{No}$
x_3	No	Yes	No	No	Some	\$	No	No	Burger	0-10	$y_3 = \text{Yes}$
x_4	Yes	No	Yes	Yes	Full	\$	Yes	No	Thai	10-30	$y_4 = \text{Yes}$
x_5	Yes	No	Yes	No	Full	\$\$\$	No	Yes	French	>60	$y_5 = \text{No}$
x_6	No	Yes	No	Yes	Some	\$\$	Yes	Yes	Italian	0-10	$y_6 = \text{Yes}$
x_7	No	Yes	No	No	None	\$	Yes	No	Burger	0-10	$y_7 = \text{No}$
x_8	No	No	No	Yes	Some	\$\$	Yes	Yes	Thai	0-10	$y_8 = \text{Yes}$
x_9	No	Yes	Yes	No	Full	\$	Yes	No	Burger	>60	$y_9 = \text{No}$
x_{10}	Yes	Yes	Yes	Yes	Full	\$\$\$	No	Yes	Italian	10-30	$y_{10} = \text{No}$
x_{11}	No	No	No	No	None	\$	No	No	Thai	0-10	$y_{11} = \text{No}$
x_{12}	Yes	Yes	Yes	Yes	Full	\$	No	No	Burger	30-60	$y_{12} = \text{Yes}$

Alternate: 附近有可替代的餐馆吗.
Bar: 餐厅是否有舒适的酒吧区等待.
Fri/Sat: 是周五周六吗.
Hungry: 是否感觉到饿.
Patrons: 有多少人在餐厅 (values are None, Some, and Full).
Price: 餐厅的价格范围 (\$, \$\$, \$\$\$).
Raining: 外面是在下雨.
Reservation: 是否预定了 (whether we made a reservation).
Type: 餐厅类型 (French, Italian, Thai or Burger).
WaitEstimate: 估计等待时间 (0-10 minutes, 10-30, 30-60, >60).

16.1.1 决策树



Example 2: with Discrete Inputs

编号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
色泽	青绿	乌黑	乌黑	青绿	浅白	青绿	乌黑	乌黑	乌黑	青绿	浅白	浅白	青绿	浅白	乌黑	浅白	青绿
根蒂	蜷缩	蜷缩	蜷缩	蜷缩	蜷缩	稍蜷	稍蜷	稍蜷	稍蜷	硬挺	硬挺	蜷缩	稍蜷	稍蜷	稍蜷	蜷缩	蜷缩
敲声	浊响	沉闷	浊响	沉闷	浊响	浊响	浊响	浊响	沉闷	清脆	清脆	浊响	浊响	沉闷	浊响	浊响	沉闷
纹理	清晰	清晰	清晰	清晰	清晰	清晰	稍糊	清晰	稍糊	清晰	模糊	模糊	稍糊	稍糊	清晰	模糊	稍糊
脐部	凹陷	凹陷	凹陷	凹陷	凹陷	稍凹	稍凹	稍凹	稍凹	平坦	平坦	平坦	凹陷	凹陷	稍凹	平坦	稍凹
触感	硬滑	硬滑	硬滑	硬滑	硬滑	软粘	软粘	硬滑	硬滑	软粘	硬滑	软粘	硬滑	硬滑	软粘	硬滑	硬滑
好瓜	是	是	是	是	是	是	是	是	否	否	否	否	否	否	否	否	否

取自：周志华 著：机器学习, 清华大学出版社, 2015.

Example 2: with Discrete Inputs

例子：17个样本实例，6个特征，2个类别

色泽：{青绿、乌黑、浅白}

根蒂：{蜷缩、稍蜷、硬挺}

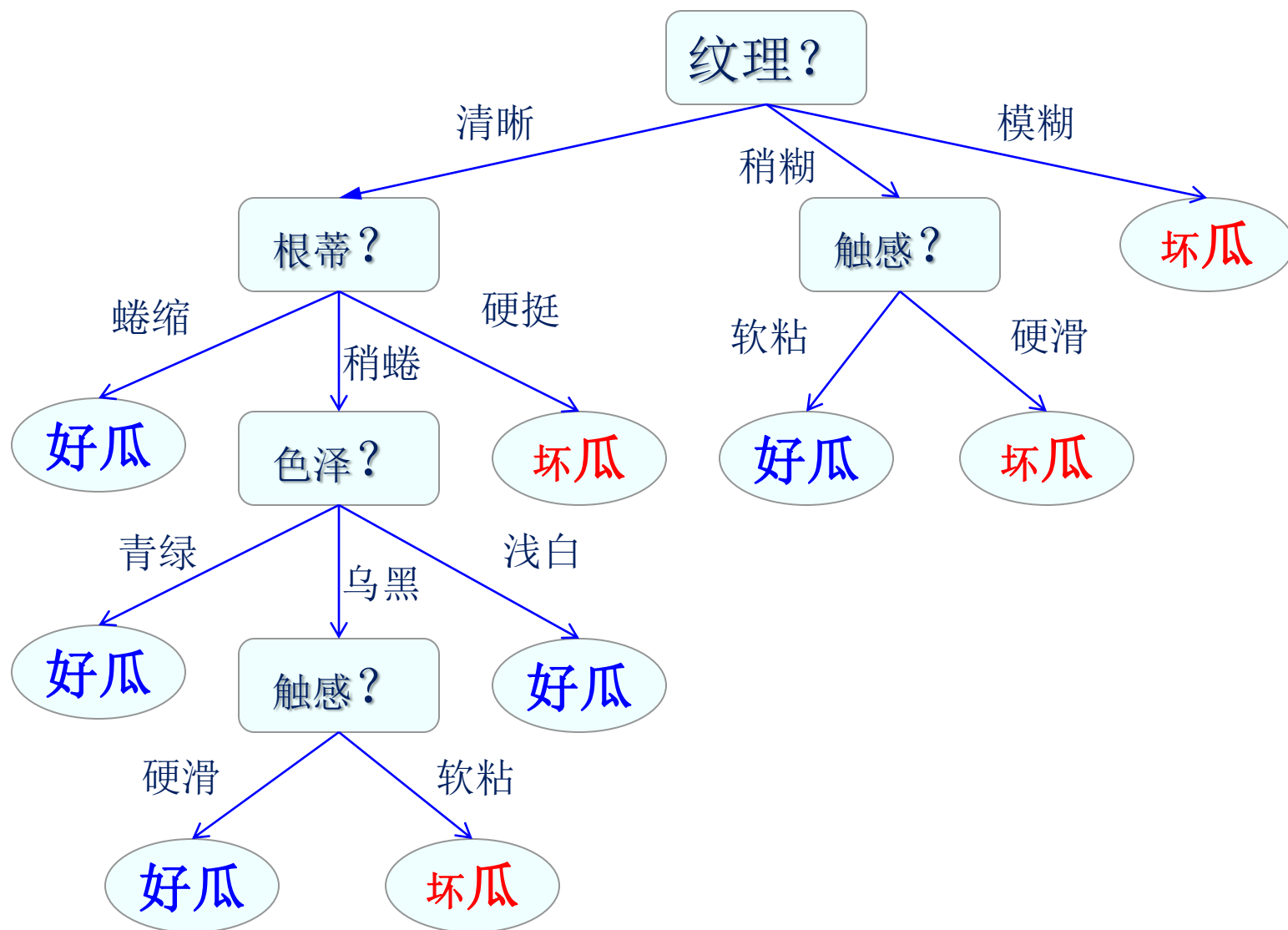
敲声：{浊响、沉闷、清脆}

纹理：{清晰、稍糊、模糊}

脐部：{凹陷、稍凹、平坦}

触感：{硬滑、软粘}

16.1.1 决策树

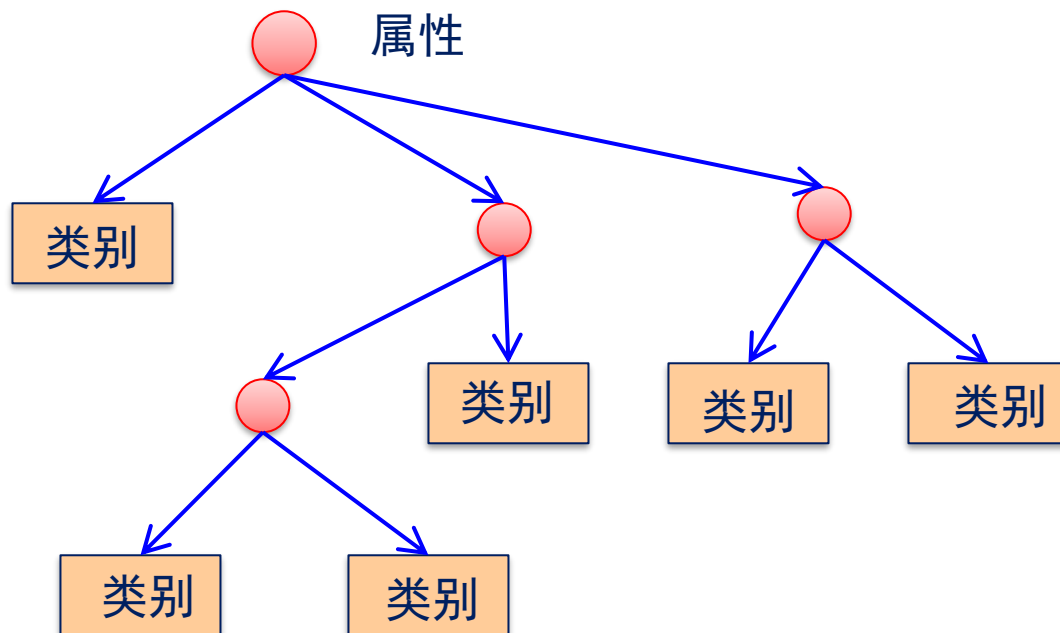


16.1.1 决策树

定义： 分类决策树是一种描述对样本进行分类的树形结构。
决策树由结点和有向边组成。

内部结点： 对应数据的一个特征（属性）

叶结点： 对应数据的一个类别

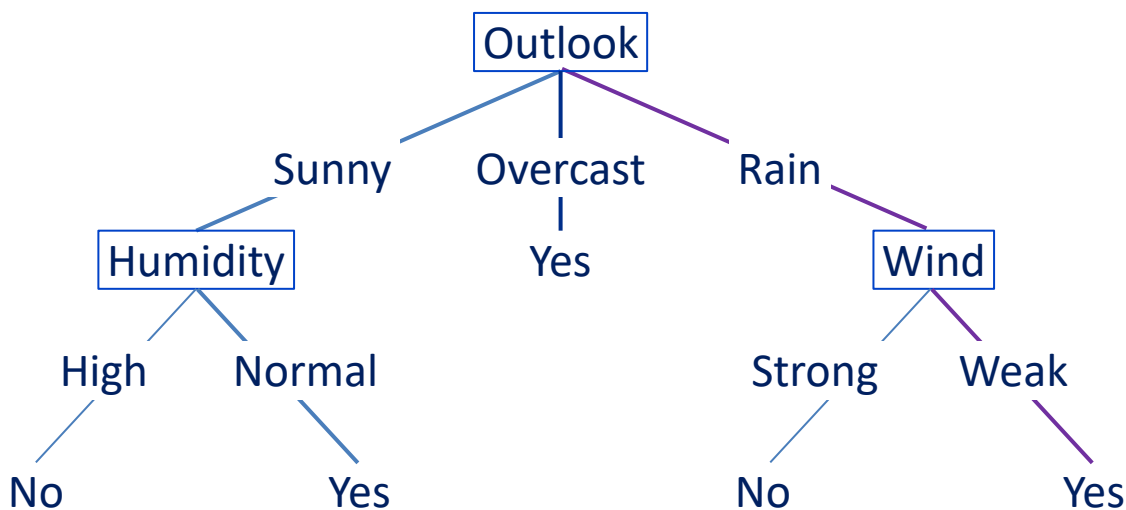


16.1 决策树

- 每一个**节点**表示对样本实例的某个属性值进行测试，该节点后相连接的各条路径（**节点的出边**）上的值表示该属性的可能取值（二叉，三叉， ...）
- 每一个**叶子**产生一条规则
 - 规则：由根节点到该叶子的路径上所有节点的条件同时成立
 - 叶子标注结论（决策，分类，判断）
- 决策树类型：分类决策树和回归决策树

16.1 决策树

- 决策树**规则**代表实例属性值约束的合取（交集）的析取（并集）式。
 - 从树根到树叶的每一条路径对应一组属性测试的合取，树本身对应这些合取的析取



(Outlook = sunny \wedge Humidity = normal)

✓ **(Outlook = overcast)**

✓ **(Outlook = rain \wedge Wind = weak)**

} Yes

16.1.1 决策树

Splitting Continuous Attributes（如何处理连续属性）：

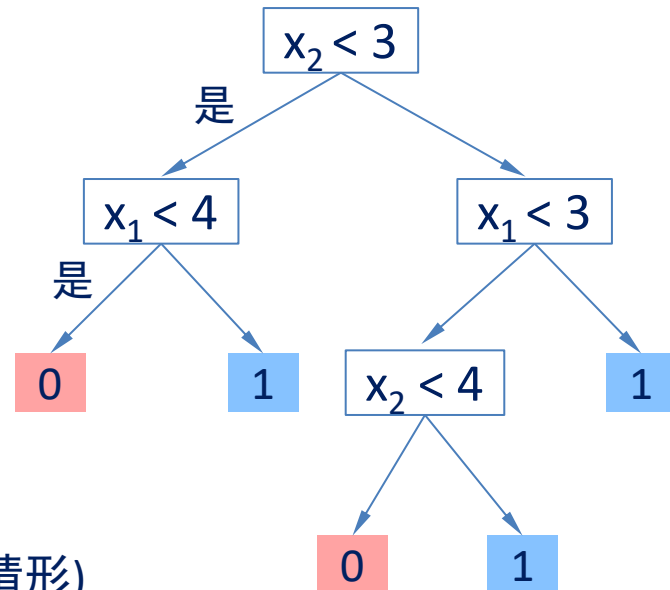
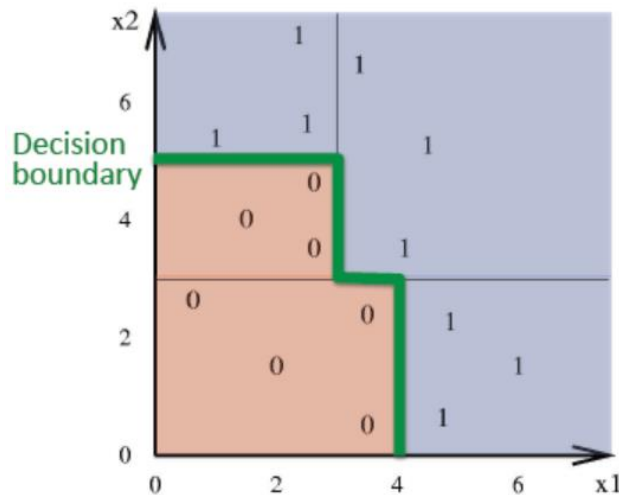
How to split continuous attributes, such as Age, Income, etc.:

- **Discretization** to form an ordinal categorical attribute
 - **Static**: discretize once at the beginning（静态离散化）
 - **Dynamic**: find ranges by equal interval bucketing (等间隔跳变), equal frequency bucketing (等频率跳变), percentiles (百分位数), clustering (聚类), etc.
- **Binary Decision**: $(A < v)$ or $(A \geq v)$
 - Consider all possible split and find the best cut
 - Often, computationally intensive (计算密集)

16.1.1 决策树

Decision Tree: Boundary

- Decision trees divide the feature space into axis-parallel (hyper-) rectangles (hyper-cubes) (将特征空间划分为轴平行区域)
- Each hyper-cube is identified with one label
 - or a probability distribution over labels
- Each path from root to a leaf defines a region R of the input space, where we let the samples with the same label fall into this region

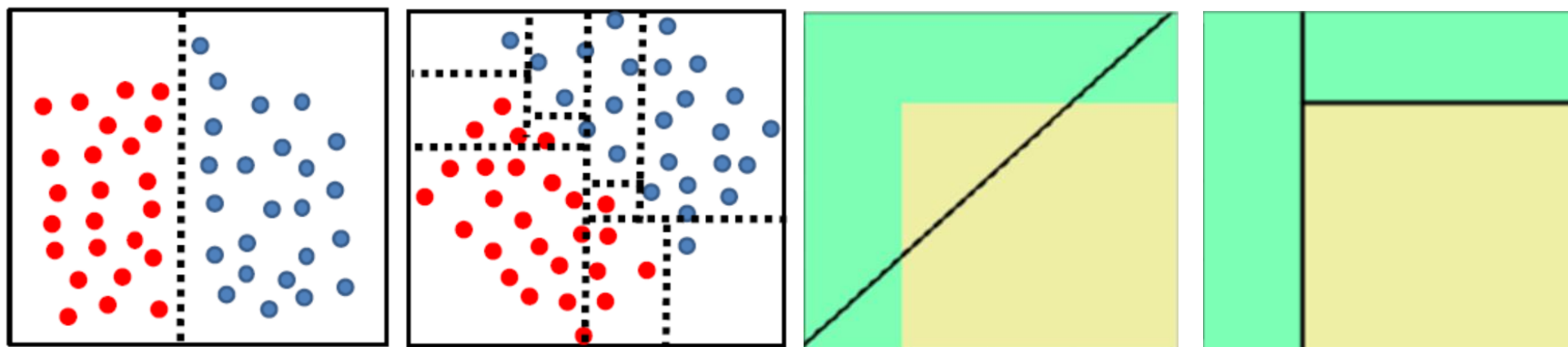


(连续情形)

16.1.1 决策树

Decision Tree: Boundary

- Tree complexity is highly dependent on data geometry in the feature space



树的复杂程度取决于数据分布的复杂度

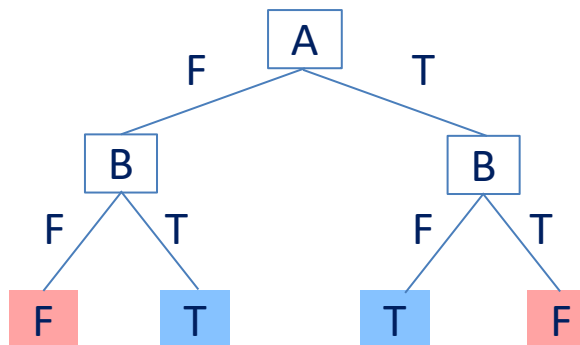
16.1.1 决策树

- 构造决策树的一般技术路线：
 - 决策树生成：遍历所有可能的特征属性，对树进行分支
 - 剪枝：适当剪除一些不必要的分支，减少过拟合，获得更好的推广能力
- 决策树涉及的三大技术：
 - 决策特征的选择
 - 决策特征分裂点的确定（特别对连续属性）
 - 决策树剪枝
- 决策树的特点：
 - 对属性采取分而治之的基本策略
 - 以树形结构，根据事先定好的“判定问题”，进行（分类）决策
 - 内部结点对应属性测试（回答“判定问题”）
 - 叶结点对应决策结果

16.1.2 Expressiveness (表达能力)

- **Discrete-input, discrete-output case:**
 - Decision trees **can express any function** of the input attributes.
 - E.g., for Boolean functions, truth table row \rightarrow path to leaf:

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



- **Continuous-input, continuous-output case:**
 - Can approximate any function arbitrarily closely (能以任意精度近似任意函数)
- Trivially, there is a consistent decision tree for any training set (one path to leaf for each example), but it probably won't generalize to new examples (**consider 1-NN**): (任何训练集都有一个一致的决策树，即每个样本都有一条通往叶子的路径)
- Need some kind of regularization to ensure more compact decision trees (需要某种正则化来确保更紧凑的决策树)

16.1.2 Expressiveness (表达能力)

Advantages and Disadvantages:

- Very easy to explain to people
 - Trees can be displayed graphically, and are easily interpreted even by a non-expert (能以图的形式来呈现, 且易于理解)
 - Some people believe that decision trees more closely mirror human decision-making (有人认为: 决策树更能反映人类的决策)
- Trees can easily handle qualitative predictors without the need to create dummy variables. (容易处理定性预测)
- Inexpensive to construct (构建成本相对较低)
- Extremely fast at classifying new data (推理快)
- Unfortunately, trees generally do not have the same level of predictive accuracy as other classifiers (精度不如其它分类器)

16.1.3 决策树学习

Learning Decision Trees

- Learning the simplest (smallest) decision tree is **an NP-complete problem** [Hyafil & Rivest'76]
 - Resort to a greedy heuristic （转而寻找启发式策略）：
 - Start from empty decision tree
 - Split on next best attribute (feature)
 - Recursive
 - ID3 (Iterative Dichotomizer)
 - C4.5 (ID3+improvements)
 - CART (Classification and Regression Trees)
- } 经典算法
- What is the best attribute?
 - We use **information theory** to guide us

16.1.3 决策树学习

给定训练数据集：

$$D = \{(x_1, y_1), \dots, (x_N, y_N)\}$$

x_i^j ：表示第 i 个样本实例的第 j 个特征（属性），离散取值。

$y_i \in \{1, 2, \dots, C\}$ ：第 i 个样本实例的类别标记

学习目标：不仅对训练数据有很好的拟合，而且对未知数据有很好的预测。

决策树的学习就是决策树的构建过程。

16.1.3 决策树学习

决策树构建的技术路线：

1. 构建根结点，将所有训练数据都放在根结点。
2. 选择一个“最优”特征，按照它的离散取值，将上一个节点中数据分割成不同子集。
3. 对于分割的子集，如果能够基本正确分类（即：子集中大部分数据都属于同一类），那么构建叶结点。
4. 对于不能构建叶结点的子集，继续2-3步，直到所有子集都能构建叶结点。

核心任务：如何选择“最优”特征，对数据集进行划分？

16.2 信息论相关知识

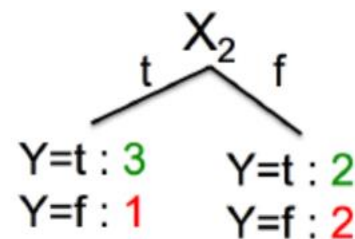
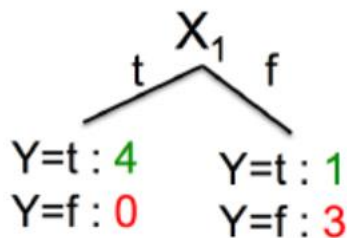
- ✓ 决策树划分的目的：每一个分支结点包含的样本尽可能属于同一类别，即分支结点的“纯度”尽可能高。
- ✓ 因此，给定一个特征对数据集合进行划分，如何评价划分后子集的“纯度”？
 - 基于信息熵的判据
 - 基于基尼指数的判据
 - 基于增益率
 - 基于最小平方误差

16.2.1 Entropy

Choosing a good attribute

- Which attribute is better to split on? (哪一个特征更具有类别区分性)
- Use counts at leaves to define probability distributions, so we can measure **uncertainty** (使用叶子的计数来定义概率分布, 以测量不确定性)

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F



16.2.1 Entropy

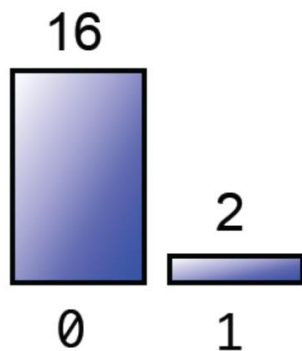
Flip Two Different Coins (翻转两枚不同的硬币)

Sequence 1:

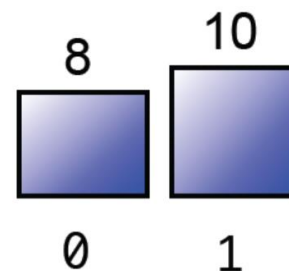
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 ... ?

Sequence 2:

0 1 0 1 0 1 1 1 0 1 0 0 1 1 0 1 0 1 ... ?

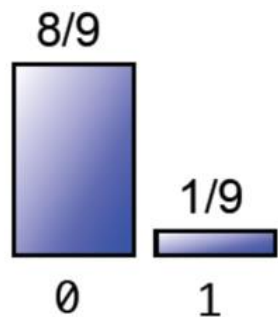


versus



16.2.1 Entropy

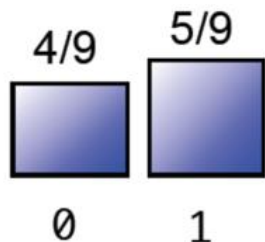
$$H(X) = - \sum_{x \in X} p(x) \log_2 p(x)$$



$$-\frac{8}{9} \log_2 \frac{8}{9} - \frac{1}{9} \log_2 \frac{1}{9} \approx \frac{1}{2}$$

“Low Entropy”： 相对容易预测

- ✓ Distribution of variable has many peaks and valleys.
(有很多高峰、低谷)
- ✓ Histogram has many lows and highs.
- ✓ Values sampled from it are more predictable.



$$-\frac{4}{9} \log_2 \frac{4}{9} - \frac{5}{9} \log_2 \frac{5}{9} \approx 0.99$$

“High Entropy”： 相对难以预测

- ✓ Variable has a uniform like distribution (平坦均匀)
- ✓ Flat histogram.
- ✓ Values sampled from it are less predictable.

熵与不确定性

在信息论与概率统计中，熵(entropy)是表示随机变量不确定性的度量。设 X 是一个取有限个值的离散随机变量，其概率分布为

$$P(X = x_i) = p_i, i = 1, 2, \dots, n$$

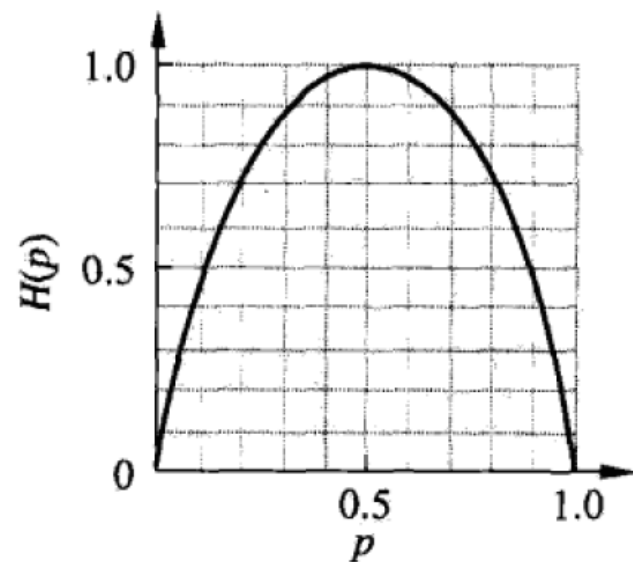
则随机变量 X 的熵 定义为
$$H(X) = - \sum_{i=1}^n p_i \log p_i$$

当随机变量只取两个值 (如1, 0) 时，即 X 的分布为

$$P(X = 1) = p, P(X = 0) = 1 - p, 0 \leq p \leq 1$$

此时，熵为

$$H(p) = -p \log_2 p - (1 - p) \log_2 (1 - p)$$



贝努利分布熵与概率的关系

16.2.2 条件熵

设有随机变量 (X, Y) ，其联合概率分布为

$$P(X = x_i, Y = y_j) = p_{ij}, \quad i = 1, 2, \dots, n; \quad j = 1, 2, \dots, m$$

条件熵 $H(Y/X)$ ：表示在已知随机变量 X 的条件下随机变量 Y 的不确定性。

条件熵(conditional entropy) $H(Y/X)$ ，定义为 X 给定条件下 Y 的条件概率分布的熵对 X 的数学期望。对离散情形，我们有：

$$H(Y/X) = \sum_{i=1}^n p_i H(Y/X = x_i) \quad (\text{注: 后面也记为 } Ent(D))$$

这里， $p_i = P(X = x_i)$ ， $i = 1, 2, \dots, n$ 。

16.2.3 信息增益（互信息）

信息增益(information gain)：表示在考察某特征信息 X 已知时，使“类 Y 的信息”的不确定性减少的程度。

特征 A 对训练数据集 D 的信息增益 $g(D, A)$ ：定义为集合 D 的熵 $H(D)$ 与 特征 A 给定条件下 D 的条件熵 $H(D|A)$ 之差，即

$$g(D, A) = H(D) - H(D|A) \quad (\text{注: 后面也记为 } \text{Gain}(D))$$

- ✓ 一般地，熵 $H(Y)$ 与条件熵 $H(Y|X)$ 之差称为互信息 (mutual information)。
- ✓ 决策树学习中的信息增益等价于训练数据集中 类与特征的互信息。
- ✓ 基于信息增益准则的特征选择：对训练数据集 (或子集) D ，计算其每个特征的信息增益，并比较它们的大小，选择信息增益最大的特征。

16.2.3 信息增益（互信息）

算法：信息增益计算

输入：训练数据集 D 和特征 A ;

输出：特征 A 对训练数据集 D 的信息增益 $g(D, A)$.

(1): 计算数据集 D 的经验熵 $H(D)$

$$H(D) = - \sum_{k=1}^K \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|}$$

(2): 计算特征 A 对数据集 D 的经验条件熵 $H(D|A)$

$$H(D|A) = \sum_{i=1}^n \frac{|D_i|}{|D|} H(D_i) = - \sum_{i=1}^n \frac{|D_i|}{|D|} \sum_{k=1}^K \frac{|D_{ik}|}{|D_i|} \log_2 \frac{|D_{ik}|}{|D_i|}$$

(3): 计算信息增益

$$g(D, A) = H(D) - H(D|A)$$

信息增益计算举例

对下表所给的训练数据集D，根据信息增益准则选择最优特征

是否拖欠贷款

贷款申请样本数据表

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

信息增益计算举例

解： 首先，计算经验熵 $H(D)$.

$$H(D) = -\frac{9}{15} \log_2 \frac{9}{15} - \frac{6}{15} \log_2 \frac{6}{15} = 0.971 \quad (\text{两类分类问题})$$

然后，计算各特征对数据集 D 的信息增益。分别以 A_1, A_2, A_3, A_4 表示年龄、有工作、有自己的房子和信贷情况等4个特征。则

$$\begin{aligned} (1) \quad g(D, A_1) &= H(D) - \left[\frac{5}{15} H(D_1) + \frac{5}{15} H(D_2) + \frac{5}{15} H(D_3) \right] \\ &= 0.971 - \left[\frac{5}{15} \left(-\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} \right) + \frac{5}{15} \left(-\frac{3}{5} \log_2 \frac{3}{5} - \right. \right. \\ &\quad \left. \left. \frac{2}{5} \log_2 \frac{2}{5} \right) + \frac{5}{15} \left(-\frac{4}{5} \log_2 \frac{4}{5} - \frac{1}{5} \log_2 \frac{1}{5} \right) \right] \\ &= 0.971 - 0.888 = 0.083 \end{aligned}$$

年龄

年龄属性=青年，共2个
标签=“是”

年龄属性=中年
标签=“是”

这里， D_1, D_2, D_3 分别是 D 中 A_1 (年龄)取值为青年、中年和老年的样本子集。类似地，

信息增益计算举例

$$\begin{aligned}(2) \quad g(D, A_2) &= H(D) - \left[\frac{5}{15} H(D_1) + \frac{10}{15} H(D_2) \right] \\ &= 0.971 - \left[\frac{5}{15} \times 0 + \frac{10}{15} \left(-\frac{4}{10} \log_2 \frac{4}{10} - \frac{6}{10} \log_2 \frac{6}{10} \right) \right] = 0.324\end{aligned}$$

$$\begin{aligned}(3) \quad g(D, A_3) &= 0.971 - \left[\frac{6}{15} \times 0 + \frac{9}{15} \left(-\frac{3}{9} \log_2 \frac{3}{9} - \frac{6}{9} \log_2 \frac{6}{9} \right) \right] \\ &= 0.971 - 0.551 = 0.420 \quad (\text{最大})\end{aligned}$$

$$(4) \quad g(D, A_4) = 0.971 - 0.608 = 0.363$$

最后，比较各特征的信息增益值，由于特征 A_3 (有自己的房子)的信息增益值最大，所以选择特征 A_3 作为最优特征。

16.2.4 信息增益率

信息增益对取值个数较多的属性有偏好，相应的增益率会大。

信息增益率/增益比：利用数据集D关于属性a的熵对信息增益进行归一化。

$$Gain_ratio(D, a) = \frac{g(D, a)}{H(D | a)}$$

其中，数据集D关于属性a 的熵：

$$H(D | a) = \sum_{i=1}^V \frac{|D^i|}{|D|} H(D^i) \quad (\text{注：属性可取值个数多，} H(D|a) \text{ 也会大})$$

C4.5决策树学习算法：根据**信息增益率**，确定“最优”特征，构建决策树。

信息增益率会对取值个数较少的属性有偏好。

先从候选属性中找出信息增益高于平均水平的属性，再从中选择增益率最高的。

16.2.5 基尼值与基尼指数

假定样本集合 D 中第 k 类样本所占比例为 p_k , $k=1,2,\dots,C$, 则样本集合 D 的**基尼值**定义为:

$$Gini(D) = \sum_{i=1}^C \sum_{j \neq i}^C p_i p_j = 1 - \sum_{i=1}^C p_i^2$$

基尼值越小，纯度越高。

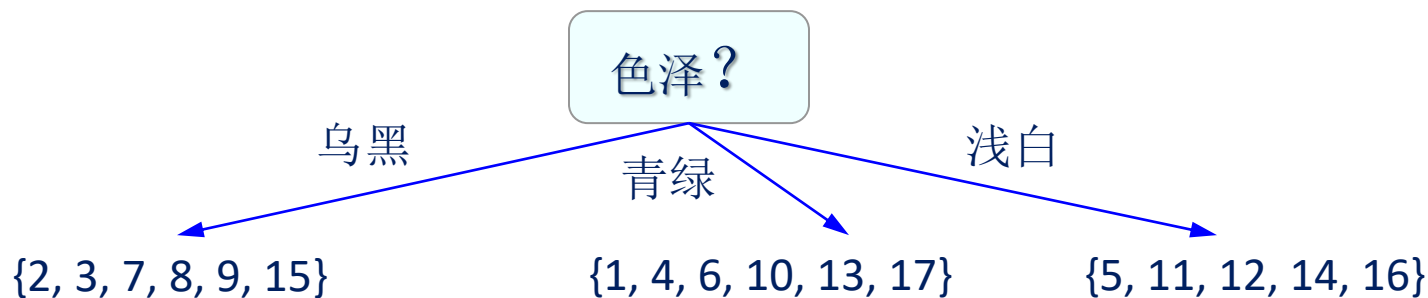
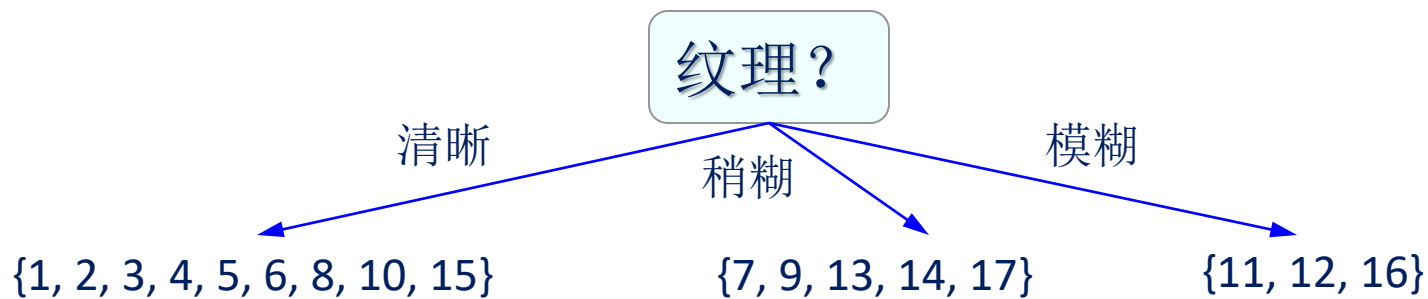
属性 a 的基尼指数：利用属性 a 对样本集合 D 划分后的加权基尼值之和。

$$Gini_index(D, a) = \sum_{i=1}^v \frac{|D^i|}{|D|} Gini(D^i)$$

16.3 决策树生成

16.3.1 (决策)特征选择

考虑对根结点进行划分：哪一个更好？



16.3.1 (决策)特征选择

- ✓ 决策树划分的目的：每一个分支结点包含的样本尽可能属于同一类别，即**分支结点的“纯度”尽可能高**。
- ✓ 给定一个特征对数据集合进行划分，**如何评价划分后子集的“纯度”**？
 - 基于信息熵的判据
 - 基于基尼指数的判据
 - 基于增益率（后面讲）

16.3.1 (决策)特征选择—回顾信息增益

假定样本集合 D 中第 k 类样本所占比例为 p_k , $k=1,2,\dots,C$, 则样本集合 D 的**信息熵**为:

$$Ent(D) = - \sum_{k=1}^C p_k \log_2 p_k \quad (\text{注意与前面换了一个符号: } H \rightarrow Ent)$$

✓ **信息熵的值越小**, 样本集的不确定性越低, D 的**纯度越高**。

信息增益: 得到样本某个属性 a 的信息, 使得样本集合的**不确定性减少的程度**。

条件信息熵: 给定样本某个属性取值的前提下, 样本集合的不确定性。

假定属性 a 有 V 个可能的离散取值 $\{a^1, a^2, \dots, a^V\}$, 若使用 a 来对样本集 D 进行划分, 则会产生 V 个子集, 其中第 i 个子集包含 D 中所有在属性 a 上取值为 a^i 的样本, 记为 D^i 。

根据属性 a , 对 D 进行划分, 整个集合的信息熵变为:

$$Ent(D|a) = \sum_{i=1}^V \frac{|D^i|}{|D|} Ent(D^i) \quad (\text{注意与前面换了一个符号: } H \rightarrow Ent)$$

根据属性 a , 对 D 进行划分所获得的**信息增益**: $Gain(D, a) = Ent(D) - Ent(D|a)$
(注意与前面换了一个符号: $g \rightarrow Gain$)

✓ **信息增益越大**, 数据集 D 划分后, “纯度”提升越大

根结点包含所有样本，8个好瓜，9个坏瓜

$$Ent(D) = -(\frac{8}{17} \log_2 \frac{8}{17} + \frac{9}{17} \log_2 \frac{9}{17}) = 0.998$$

色泽：{青绿、乌黑、浅白}

D^1 (色泽=青绿)：{1,4,6,10,13,17}

$$Ent(D^1) = -(\frac{3}{6} \log_2 \frac{3}{6} + \frac{3}{6} \log_2 \frac{3}{6}) = 1$$

D^2 (色泽=乌黑)：{2,3,7,8,9,15}

$$Ent(D^2) = -(\frac{4}{6} \log_2 \frac{4}{6} + \frac{2}{6} \log_2 \frac{2}{6}) = 0.918$$

D^3 (色泽=浅白)：{5,11,12,14,16}

$$Ent(D^3) = -(\frac{1}{5} \log_2 \frac{1}{5} + \frac{4}{5} \log_2 \frac{4}{5}) = 0.722$$

$$Gain(D, \text{色泽}) = Ent(D) - \sum_{i=1}^3 \frac{|D^i|}{|D|} Ent(D^i)$$

$$= 0.998 - (\frac{6}{17} \times 1 + \frac{6}{17} \times 0.918 + \frac{5}{17} \times 0.722)$$

$$= 0.109$$

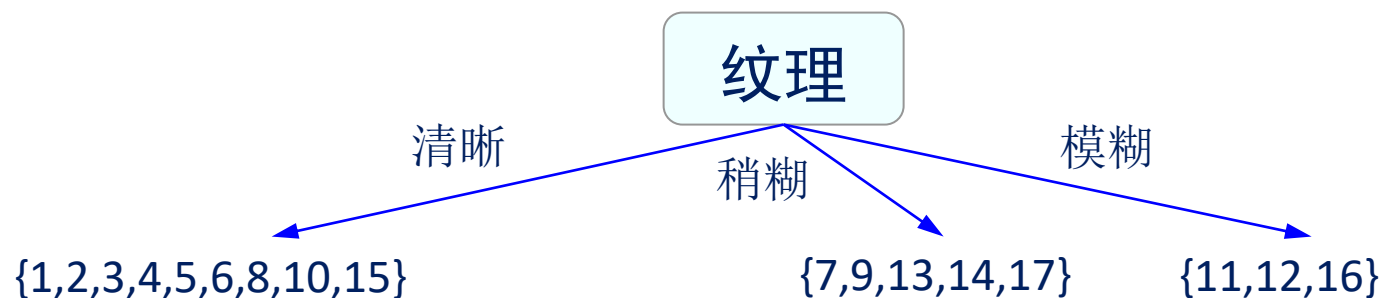
$$Gain(D, \text{根蒂}) = 0.143$$

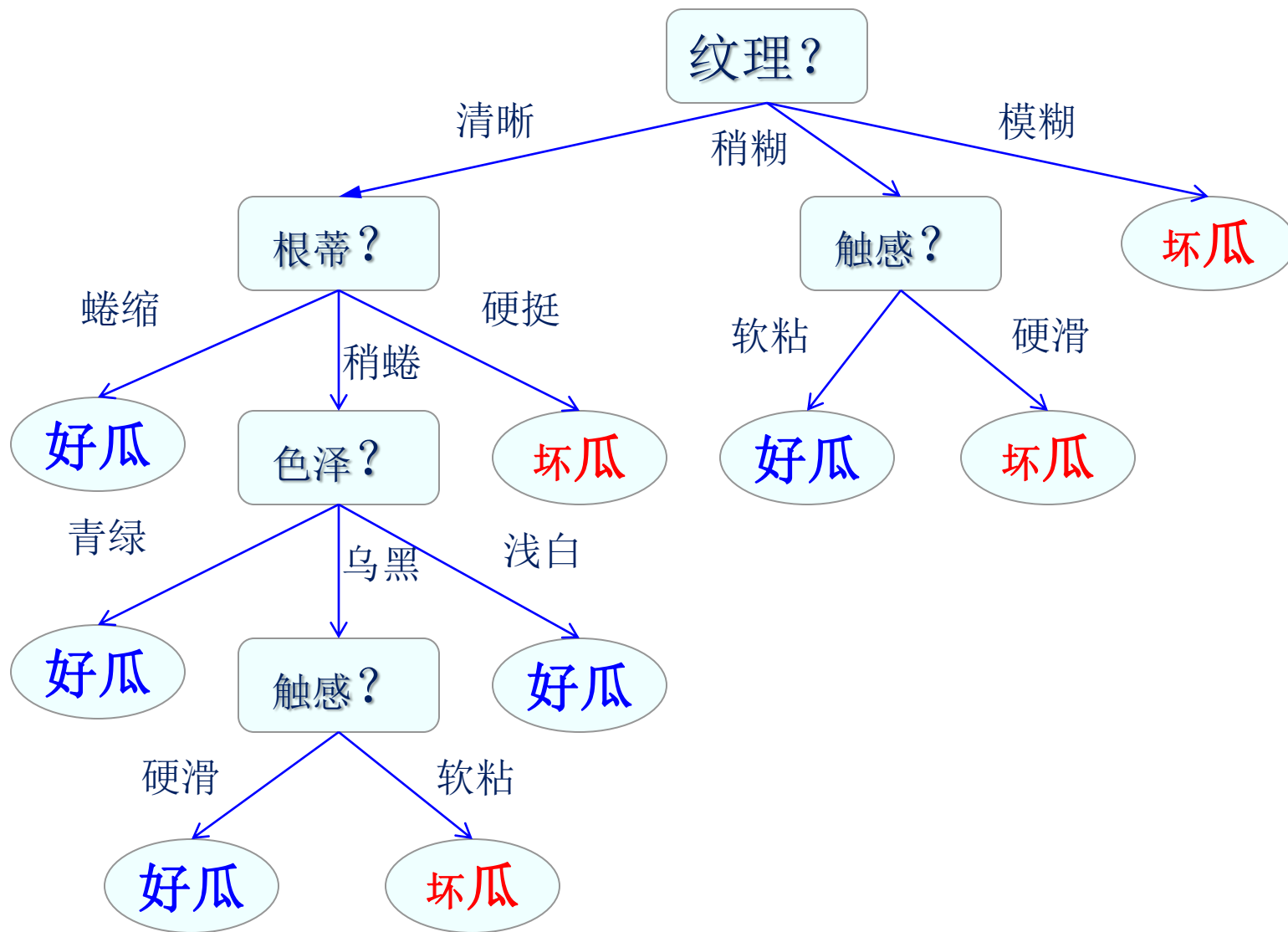
$$Gain(D, \text{敲声}) = 0.141$$

$$Gain(D, \text{纹理}) = 0.381$$

$$Gain(D, \text{脐部}) = 0.289$$

$$Gain(D, \text{触感}) = 0.006$$





16.3.1 (决策)特征选择—回顾信息增益率

信息增益对取值个数较多的属性有偏好，相应的增益率会大。

信息增益率/增益比：利用数据集D关于属性a的熵对信息增益进行归一化。

$$Gain_ratio(D, a) = \frac{Gain(D, a)}{Ent(D | a)}$$

其中，数据集D关于属性a 的熵：

$$Ent(D | a) = \sum_{i=1}^V \frac{|D^i|}{|D|} Ent(D^i) \quad (\text{注：属性可取值个数多，} Ent(D/a) \text{ 也会大})$$

C4.5决策树学习算法：根据**信息增益率**，确定“最优”特征，构建决策树。

信息增益率会对取值个数较少的属性有偏好。

先从候选属性中找出信息增益高于平均水平的属性，再从中选择增益率最高的。

16.3.2 决策树生成算法

ID3算法

输入：训练数据集 D 、特征集合 A 、阈值 t

输出：决策树 T

1. 若 D 中所有实例都属于同一类 k ，则 T 为单结点树（或叶结点），将类 k 作为该结点的类标记，返回 T ；
2. 若特征集合 A 为空集，则 T 为单结点树（或叶结点），将 D 中实例最多的类 k 作为该结点的类标记，返回 T ；
3. 否则，计算 A 中各特征对 D 的信息增益，选择信息增益最大的特征 A_m ；如果 A_m 的信息增益小于阈值 t ，则 T 为单结点树（叶结点），将 D 中实例最多的类 k 作为该结点的类标记，返回 T ；
4. 否则，对 A_m 的每一可能取值 a_i ，根据 $A_m = a_i$ 将 D 分割成非空子集 D^i ，每个子集作为一个子结点（内部结点），由输入结点及子结点构成树 T ，返回 T ；
5. 对第 i 子结点，以 D^i 为训练集， $A - \{A_m\}$ 为特征集，递归调用 ID3 算法

16.3.2 决策树生成算法

C4.5算法

输入：训练数据集 D 、特征集合 A 、阈值 t

输出：决策树 T

1. 若 D 中所有实例都属于同一类 k ，则 T 为单结点树（叶结点），将类 k 作为该结点的类标记，返回 T ；
2. 若特征集合 A 为空集，则 T 为单结点树（叶结点），将 D 中实例最多的类 k 作为该结点的类标记，返回 T ；
3. 否则，计算 A 中各特征对 D 的信息增益和信息增益率，先找出信息增益高于平均水平的特征，再从中选择增益率最大的特征 A_m ；如果 A_m 的信息增益率小于阈值 t ，则 T 为单结点树（叶结点），将 D 中实例最多的类 k 作为该结点的类标记，返回 T ；
4. 否则，对 A_m 的每一可能取值 a_i ，根据 $A_m = a_i$ 将 D 分割成非空子集 D^i ，每个子集作为一个子结点（内部结点），由输入结点及子结点构成树 T ，返回 T ；
5. 对第 i 个子结点，以 D^i 为训练集， $A - \{A_m\}$ 为特征集，递归调用 C4.5 算法

16.3.2 决策树生成算法

ID3算法

- ID3是Quinlan于1986年提出的，开创了决策树算法的先河，是国际上最早的决策树方法。
 - 在该算法中，引入了信息论中熵的概念，利用分割前后的熵来计算信息增益，作为判别能力的度量。
 - ID3 算法的核心是在决策树各个结点上应用信息增益准则选择特征，递归地构建决策树。
 - **具体方法：**从根结点开始，对结点计算所有可能的特征的信息增益，选择信息增益最大的特征作为结点的特征，由该特征的不同取值建立子结点；再对子结点递归地调用以上方法，构建决策树；直到所有特征的信息增益均很小或没有特征可以选择为止。最后得到一个决策树。
 - ID3相当于用极大似然法进行概率模型的选择。

16.3.2 决策树生成算法

C4.5算法

- C4.5算法是一种分类决策树算法，其核心算法是ID3算法，C4.5继承了ID3算法的优点，并在以下几方面对ID3算法进行了改进：
 - 用**信息增益率**来选择属性，克服了用**信息增益**选择属性时偏向选择取值多的属性的不足
 - 在树构造过程中进行剪枝
 - 能够完成对连续属性的离散化处理
 - 能够对不完整数据进行处理

16.3.2 决策树生成算法

C4.5 vs ID3

- C4.5中使用信息增益率(Gain ratio)，ID3算法使用信息增益(Info Gain)
 - 一个属性的信息增益越大，表明该属性对样本的熵变化（减少）的适应能力更强。该属性使得数据由不确定性变成确定性的能力越强。
- 信息增益在面对类别较少的离散数据时效果较好，在每个类别都有一定数量的样本的情况下，使用ID3与C4.5的区别并不大。
- 但如果面对**连续的数据**（如体重、身高、年龄、距离等），或者每列数据（对应单个属性）没有明显的类别之分，**此时，ID3算法倾向于将每个数据自成一类**（将每一个样本都分到一个节点当中去），程序会倾向于选择这种划分，但这样划分效果极差。
- 为了解决这个问题，引入了信息增益率的概念，**减轻了划分行为本身的影响**。

16.3.2 决策树生成算法

C4.5 vs ID3

- 对取值更多的属性，更容易使得数据更“纯”（尤其是连续型数值），其信息增益更大，决策树会首先挑选这个属性作为树的顶点。
 - 采取信息增益方式，若某属性取值数目远大于类别数量时信息增益可以变得很大，对训练集分割尽管不错，但泛化能力同时也会变差。这种分割方式导致在每个分割空间内数据较纯净，甚至能使熵为0，但未利用其它实际上可能更加有效的分类信息。
- 如果结果训练出来的形状是一棵庞大且深度很浅的树，该划分是极不合理的。
- C4.5 使用了信息增益率，在信息增益的基础上除了一项 split information (即条件熵, $Ent(D|a)$), 来惩罚值更多的属性。
 - 信息增益率引入了分裂信息，取值数目多的属性分裂信息也会变大，将增益除以分裂信息，可以有效缓解信息增益过大的问题。

16.4 决策树剪枝

16.4 决策树剪枝

Many kinds of **noise** can occur in the examples

- **Erroneous** training data (错误样本的类型是多样的)
 - two examples have same attribute/value pairs, but **different classifications**
 - **feature noise**
 - some values of attributes are incorrect because of errors in data acquisition process or preprocessing phase
 - **label noise**
 - instance was labeled incorrectly (+ instead of -)

Overfitting

Much more significant sources of “noise”

- Some attributes are **irrelevant** to decision making process （一些属性与决策无关）
 - if hypothesis space has many dimensions (large # of attributes), may find **meaningless regularity** in data that is irrelevant to true, important, distinguishing features
- Target function is **non-deterministic** in attributes （目标函数值可能并不是属性值唯一确定的）
 - in general, we cannot measure all variables needed to do perfect prediction → target function is not uniquely determined by attribute values
 - if too little training data, even a reasonable hypothesis will “overfit”

How can we avoid overfitting?

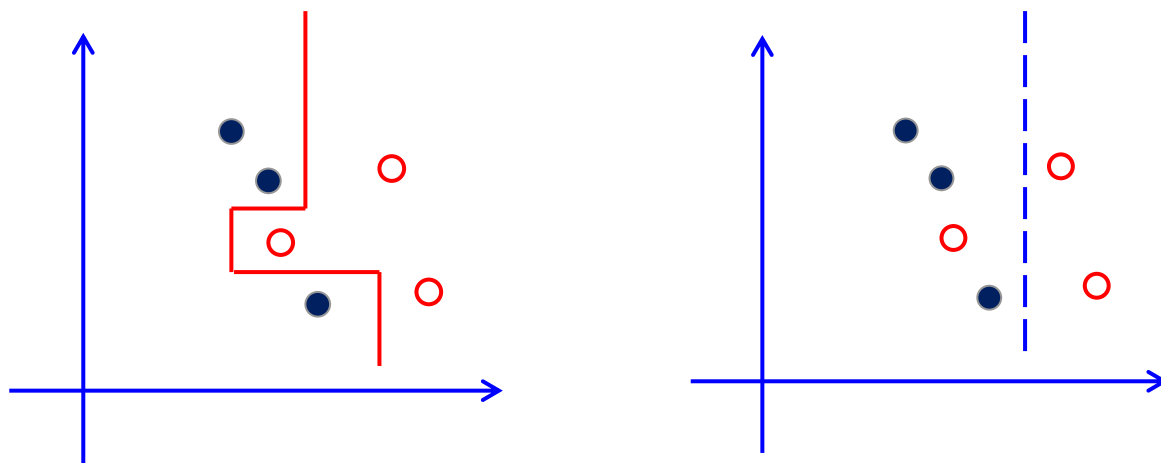
- Acquire more training data （增加训练样本）
- Remove irrelevant attributes (manual process – not always possible) （去掉不相关属性）

16.4 决策树剪枝

ID3算法和C4.5算法采用递归策略来产生决策树，直到不能继续下去为止，**容易产生过拟合**：

- 学习过程过多地考虑了如何提高分类正确率，忽略了推广性，构建出**过于复杂**的决策树
- 对训练数据很准确，对未知测试数据的分类性能很差

解决方法：剪枝！降低决策树复杂度。在模型复杂度和对训练数据的拟合之间进行平衡。



16.4 决策树剪枝

核心问题：如何判断决策树泛化性能有提升？

- 留出法：从带标签的训练数据中划分一部分作为“验证集”，通过在验证集上评估训练模型的分类精度，判断模型的泛化能力！

根据剪枝处于的阶段不同：

- 预剪枝：在生成决策树的过程进行剪枝，剪枝和树构建同时进行。
 - 在决策树生成过程中，对每个结点在划分前先进行估计，若对该结点的划分不会带来决策树泛化性能的提升，则不进行该划分。
- 后剪枝：先生成决策树，再进行修剪。
 - 先从训练集生成一棵完整的决策树，然后自底向上地对非叶结点进行考察，若将该结点对应的子树替换为叶结点能带来决策树泛化性能提升，则进行替换（剪枝，去掉其后的子树）。

16.4 决策树剪枝

预剪枝算法的技术路线

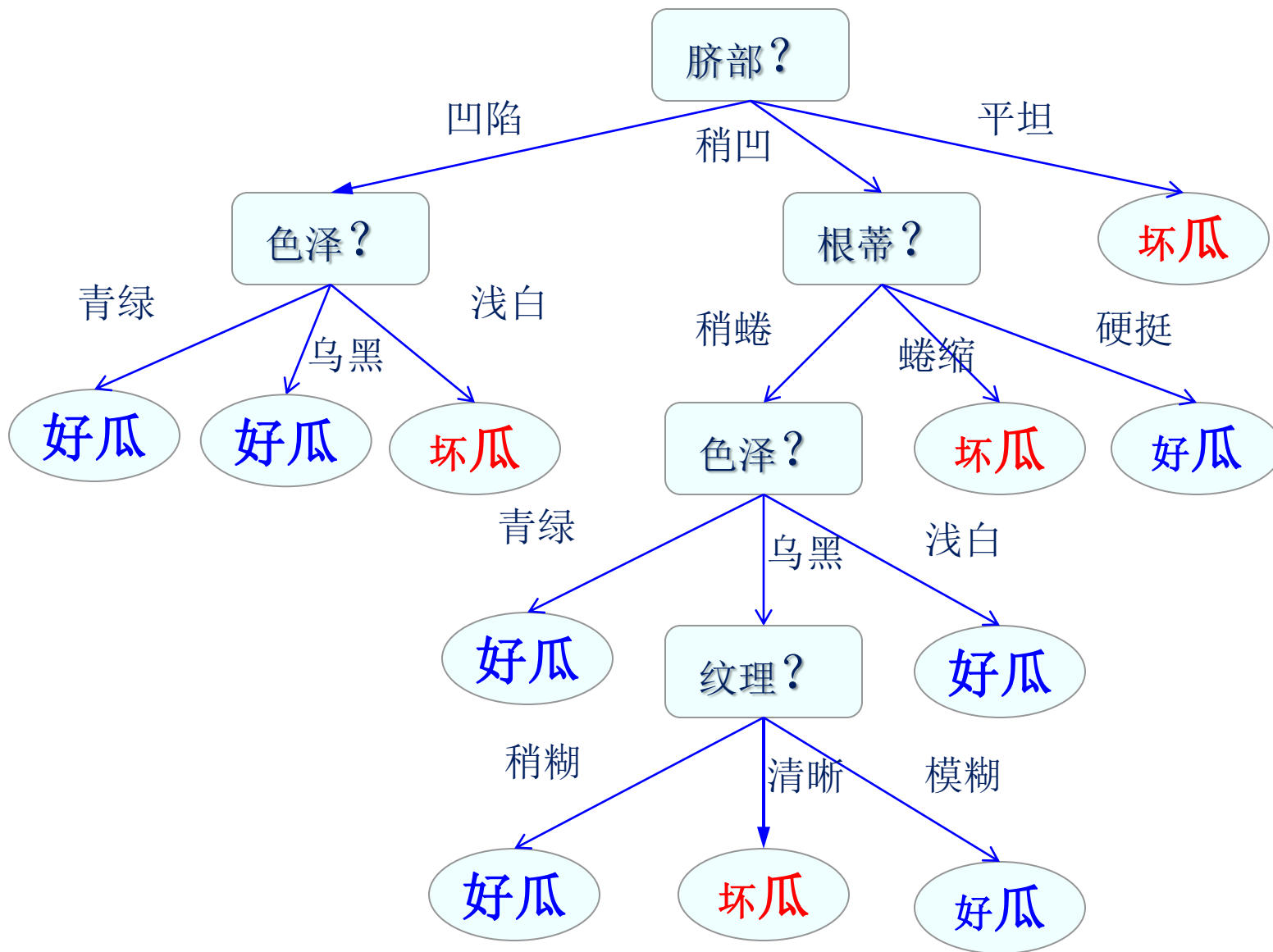
1. 在决策树生成算法中，假定选择了某个特征 a 对数据进行划分，记划分前后的决策树（划分前该结点的类别为其中训练样本中最多的类别）分别为 T_A 和 T_B 。
2. 分别计算使用 T_A 和 T_B 对验证集进行评估的性能，分类正确率记为 P_A 和 P_B 。
3. 如果 $P_B > P_A$ ，表明划分之后决策树泛化性能更好，保留该划分；否则放弃。

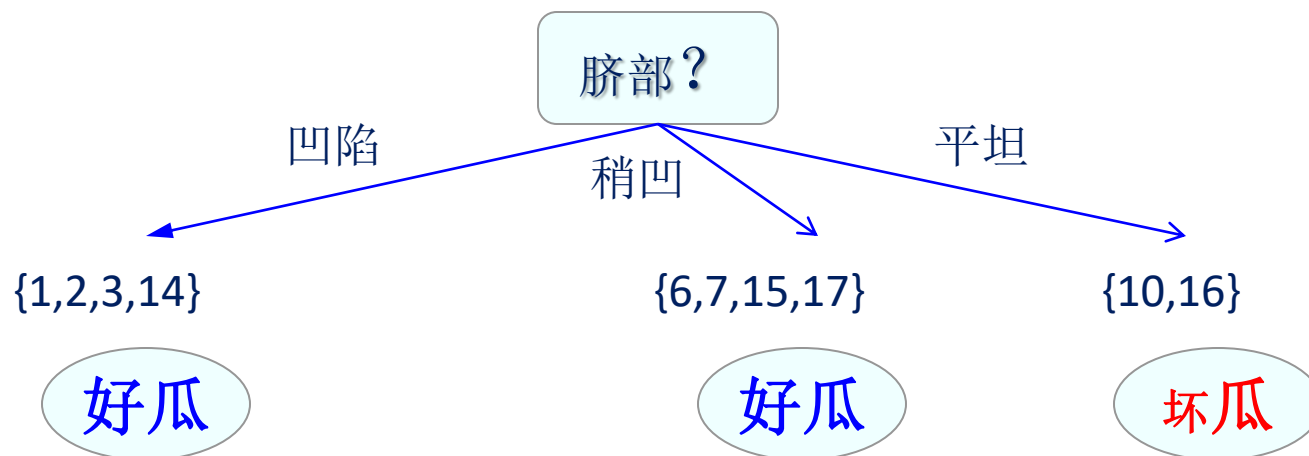
训练集（10个样本）

编号	1	2	3	6	7	10	14	15	16	17
色泽	青绿	乌黑	乌黑	青绿	乌黑	青绿	浅白	乌黑	浅白	青绿
根蒂	蜷缩	蜷缩	蜷缩	稍蜷	稍蜷	硬挺	稍蜷	稍蜷	蜷缩	蜷缩
敲声	浊响	沉闷	浊响	浊响	浊响	清脆	沉闷	浊响	浊响	沉闷
纹理	清晰	清晰	清晰	清晰	稍糊	清晰	稍糊	清晰	模糊	稍糊
脐部	凹陷	凹陷	凹陷	稍凹	稍凹	平坦	凹陷	稍凹	平坦	稍凹
触感	硬滑	硬滑	硬滑	软粘	软粘	软粘	硬滑	软粘	硬滑	硬滑
好瓜	是	是	是	是	是	否	否	否	否	否

验证集(7个样本)

4	5	8	9	11	12	13
青绿	浅白	乌黑	乌黑	浅白	浅白	青绿
蜷缩	蜷缩	稍蜷	稍蜷	硬挺	蜷缩	稍蜷
沉闷	浊响	浊响	沉闷	清脆	浊响	浊响
清晰	清晰	清晰	稍糊	模糊	模糊	稍糊
凹陷	凹陷	稍凹	稍凹	平坦	平坦	凹陷
硬滑	硬滑	硬滑	硬滑	硬滑	软粘	硬滑
是	是	是	否	否	否	否





“脐部结点”含有所有训练数据，“好瓜”、“坏瓜”各有5个，**任选一类“好瓜”**作为该结点标记类别。（因为各5个，所以如果作为叶子决策节点，则任先一类）

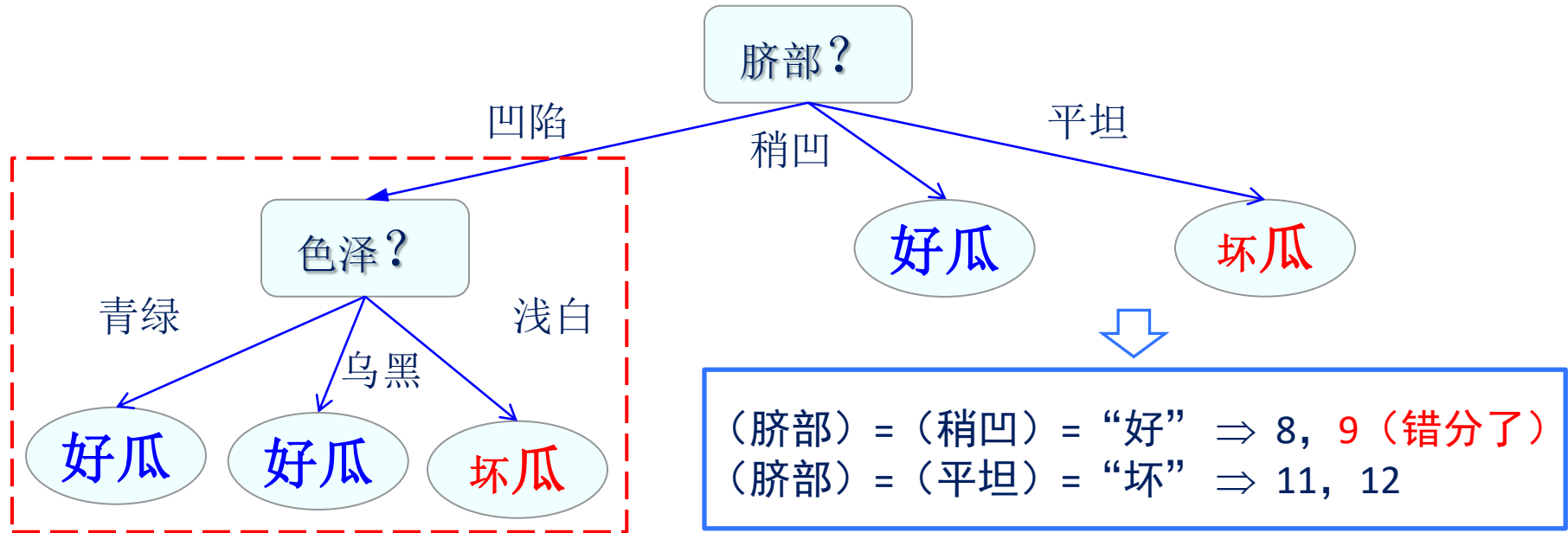
在验证集上分类正确率：3/7=42.9%

划分前

“凹陷”，“稍凹”，“平坦”三个叶结点分别表示“好瓜”，“好瓜”，“坏瓜”。

在验证集上，{4,5,8,11,12}分类正确，正确率：5/7=71.4%

划分后



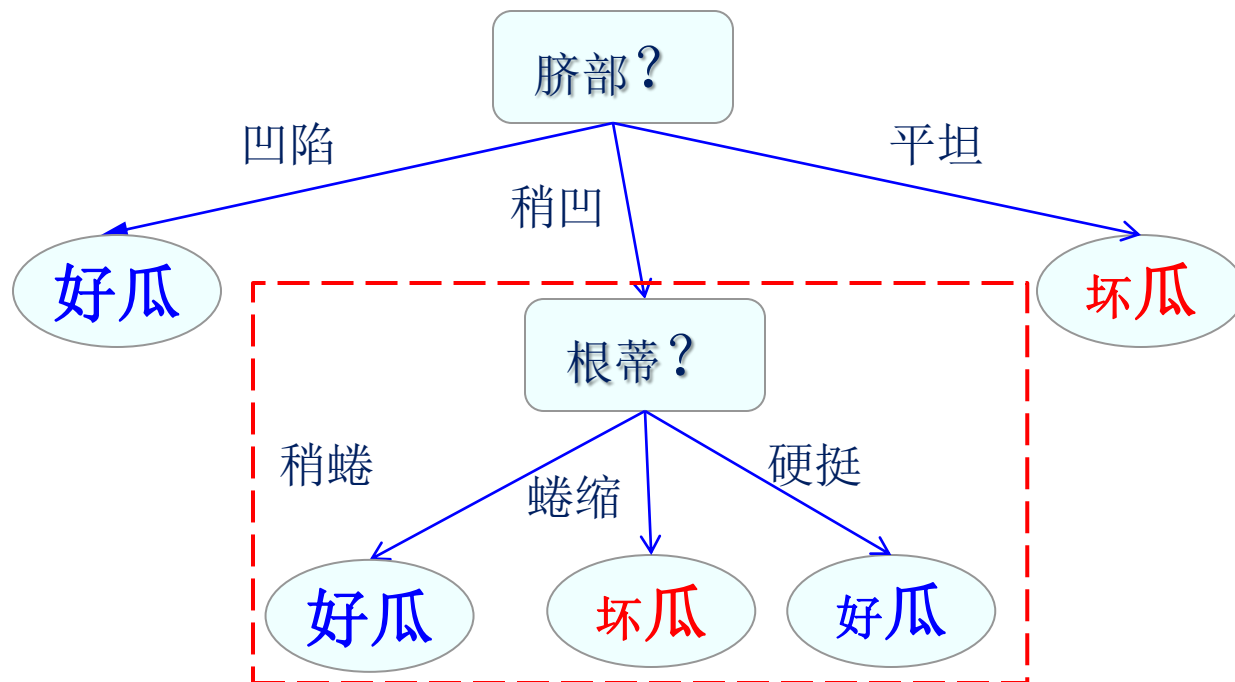
综合起来等于 $\frac{4}{7}$

(脐部, 色泽) = (凹陷, 青绿) = “好” \Rightarrow 4, 13 (错分了)
 (脐部, 色泽) = (凹陷, 乌黑) = “好” \Rightarrow 5 (错分了)
 (脐部, 色泽) = (凹陷, 浅白) = “坏” \Rightarrow 5 (错分了)

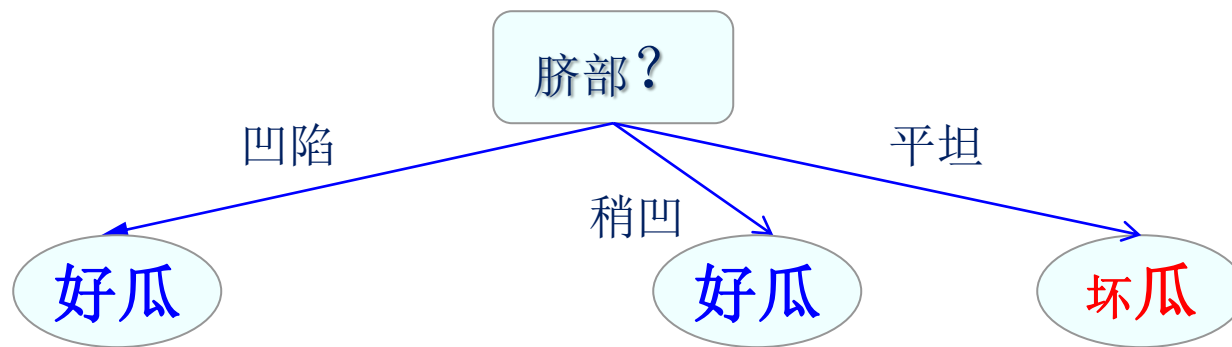
划分前的规则: “凹陷+青绿” 或 “凹陷+乌黑” = 好: 4, 13
 “凹陷+浅白” = 坏: 5 (错分了)
 (脐部) = (稍凹) = “好” \Rightarrow 8, 9 (错分了)
 (脐部) = (平坦) = “坏” \Rightarrow 11, 12

$\Rightarrow \frac{5}{7}$

划分前: $\frac{5}{7} = 71.4\%$, 划分后: $\frac{4}{7} = 57.1\%$. 禁止划分, 剪枝



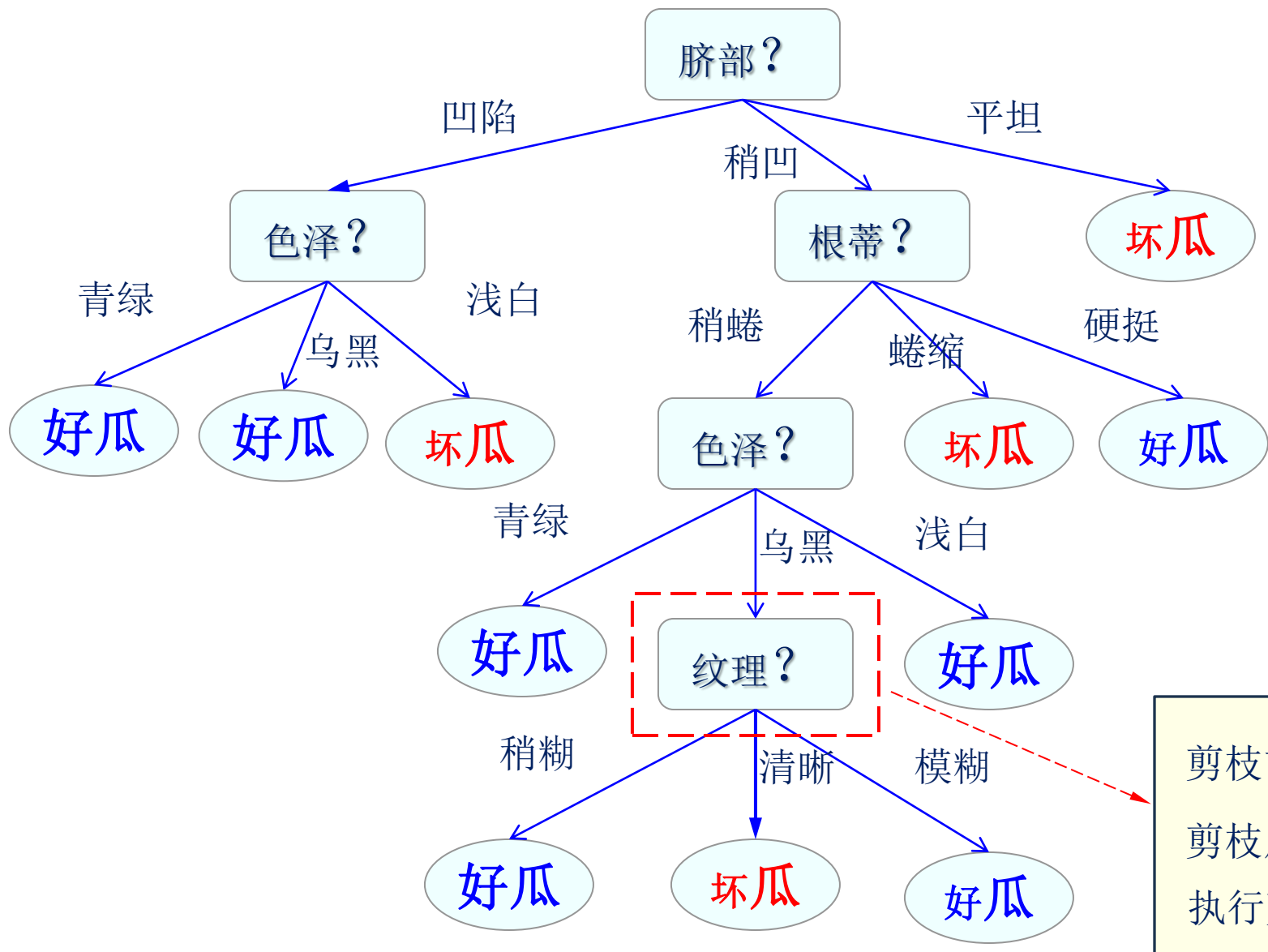
划分前：71.4%， 划分后：71.4%， 禁止划分，剪枝



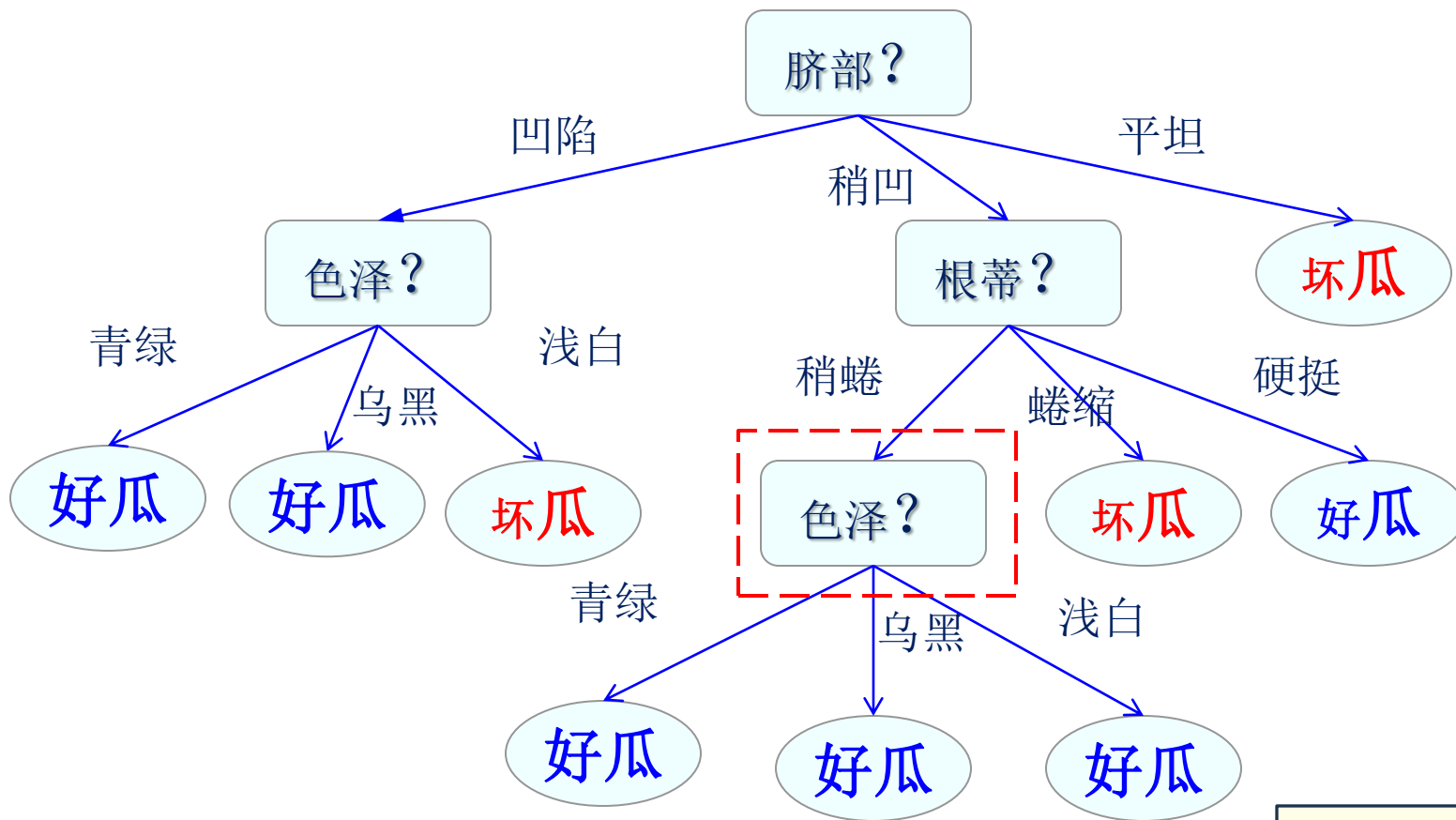
16.4 决策树剪枝

后剪枝算法的技术路线

1. 根据训练集数据，生成一棵完整的决策树。
2. 从决策树最底部的叶结点开始，向上回缩。记一组叶结点（对应同一个父结点）回缩前后，决策树在验证集上的正确率分别为 P_B 和 P_A ，如果 $P_A \geq P_B$ ，满足剪枝条件，将该组叶结点并入其父结点，并将其父结点作为新的叶结点
3. 如果 $P_A < P_B$ ，表明回缩之后决策树泛化性能更差，放弃这次剪枝。
4. 递归执行2-3步，直到不能继续为止。



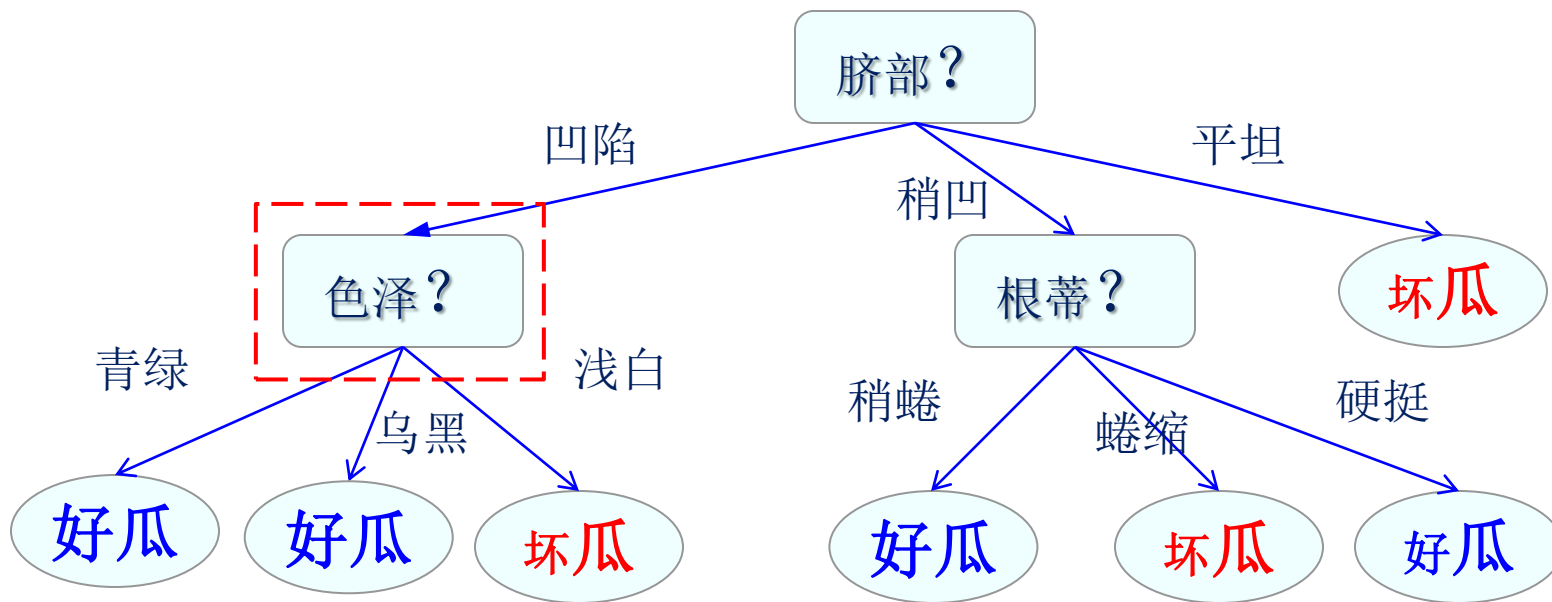
剪枝前: 42.9%
剪枝后: 57.1%
执行剪枝



剪枝前: 57.1%

剪枝后: 57.1%

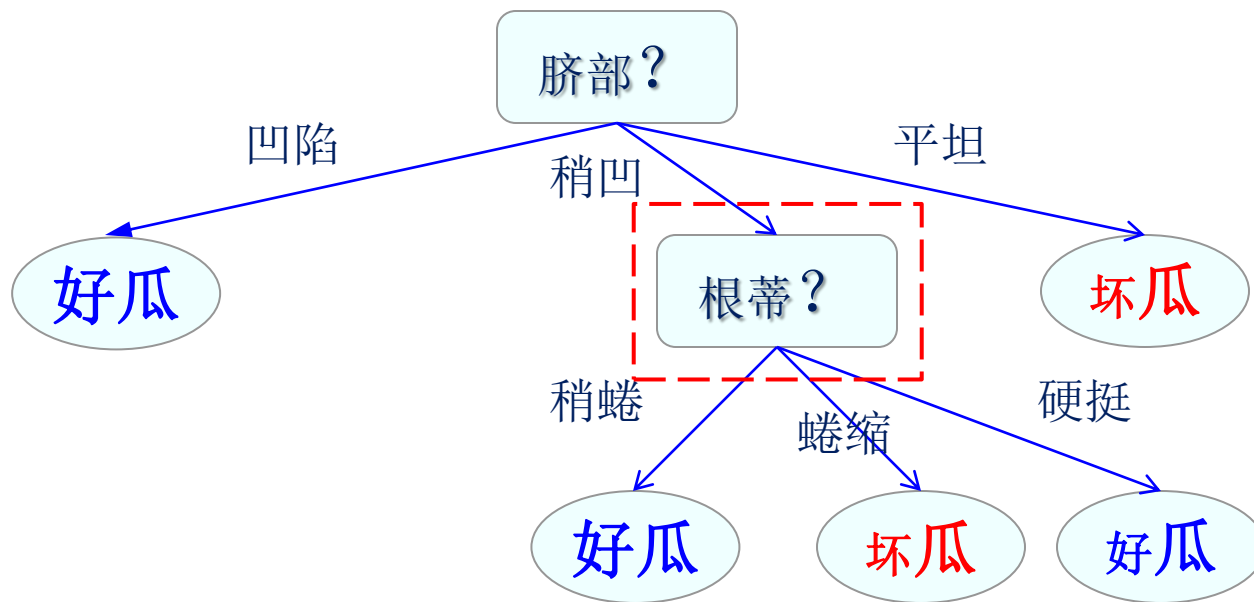
执行剪枝



剪枝前: 57.1%

剪枝后: 71.4%

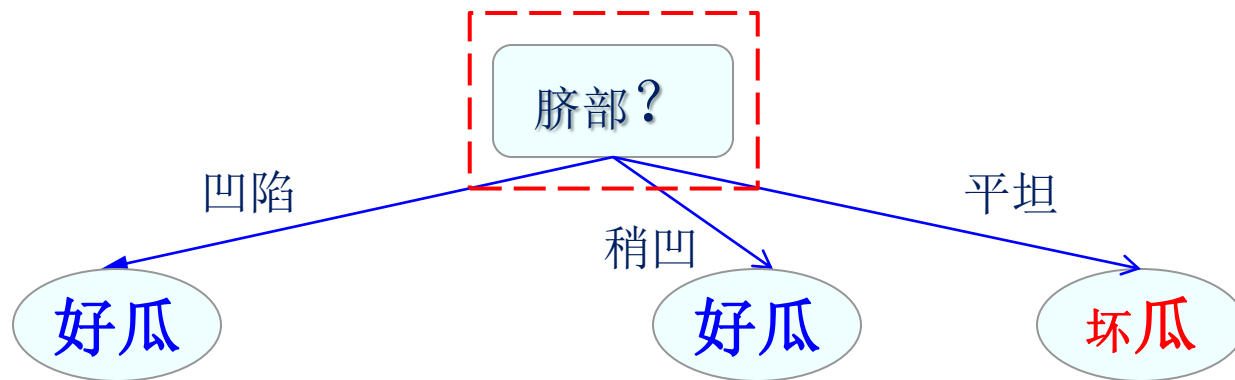
执行剪枝



剪枝前: 71.4%

剪枝后: 71.4%

执行剪枝



剪枝前: 71.4%

剪枝后: 42.9%

放弃剪枝

16.4 决策树剪枝

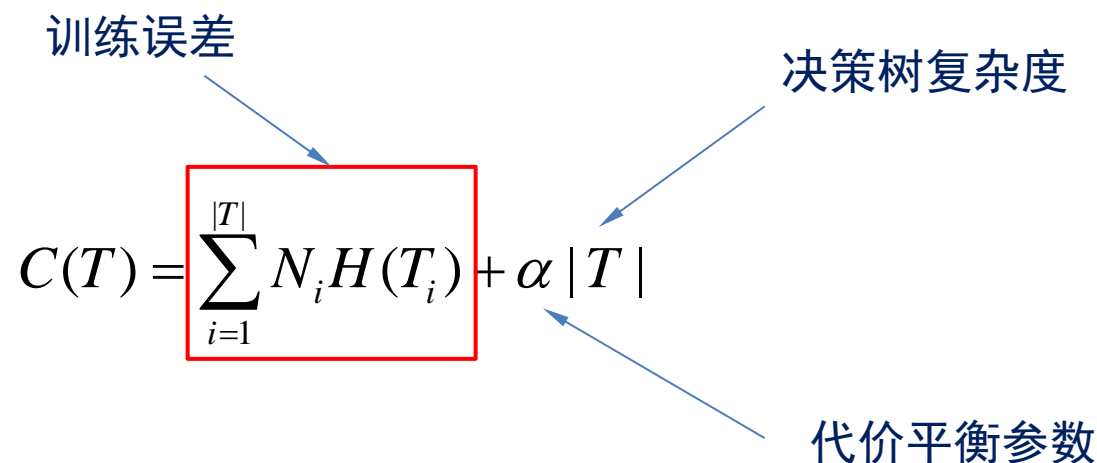
除了根据决策树泛化性能是否有提升来确定是否剪枝，还可以根据整体代价是否会减少来确实是否剪枝。

训练误差

决策树复杂度

$$C(T) = \sum_{i=1}^{|T|} N_i H(T_i) + \alpha |T|$$

代价平衡参数



其中，

$$H(T_i) = - \sum_{k=1}^c \frac{N_{ik}}{N_i} \log_2 \frac{N_{ik}}{N_i}$$

16.4 决策树剪枝

基于整体代价的决策树剪枝算法（后处理）

输入：决策树 T ，代价平衡参数 α

输出：修剪之后的决策树 T_α

1. 计算每个叶结点的信息熵
 2. 从叶结点开始向上回缩。记一组叶结点（对应同一个父结点）回缩前后，决策树的整体代价分别为 $C(T_B)$ 和 $C(T_A)$ ，如果 $C(T_A) \leq C(T_B)$ ，满足剪枝条件，将该组叶结点并入其父结点，并将其父结点作为新的叶结点。
 3. 递归调用第1-2步，直到不能继续为止。
-

16.4 决策树剪枝

缺失值处理：

不完整样本：某些属性值缺失的样本

- ① 属性值缺失的情况下，如何进行划分属性选择？
 - ② 给定划分属性，若样本在该属性上的值缺失，如何对样本进行划分？
- 为每个样本分配一个权重（通常考虑样本具有相同的权重）
 - 对于第一个问题，使用所有包含给定属性值的样本集合计算属性的信息增益/增益率/基尼指数
 - 对于第二个问题，划分至该属性的所有子结点，并根据子结点所占父结点比例调整权值

（分别依据所有数据和部分数据）

16.4 决策树剪枝

缺失值处理：

记 \tilde{D} 为样本集合 D 中属性 a 上没有缺失值的样本子集； \tilde{D}^i 表示 \tilde{D} 中在属性 a 上取值为 a^i 的样本子集；

$$Gain(D, a) = \rho \times Gain(\tilde{D}, a)$$

ρ : 无缺失值样本所占的比例, $\rho = \frac{\sum_{x_i \in \tilde{D}} w_i}{\sum_{x_i \in D} w_i}$

w_i 表示样本 x_i 的权重

16.5 连续特征决策树

16.5 连续特征决策树

离散特征：只能取固定个数的值，可依具体取值进行结点划分

连续特征：无限取值，不能根据具体特征取值对结点进行划分

连续特征离散化：通过设置一定的取值区间，将连续取值转化成离散的区间取值。常用的离散策略是**二划分**。

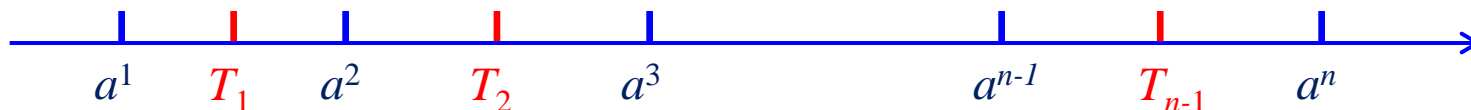
$$f(x) = \begin{cases} 0, & x > t \\ 1, & x \leq t \end{cases}$$

16.5 连续特征决策树

给定样本集 D 和连续属性 a ，假定 a 在 D 上出现了 n 个不同的取值，将这些值从小到大进行排序，记为 $\{a^1, a^2, \dots, a^n\}$ 。

基于划分点 t ，可将 D 分为 D_t^+ 和 D_t^- 两个子集，其中 D_t^+ 包含那些在属性 a 上取值大于 t 的样本，而 D_t^- 包含那些在属性 a 上取值不大于 t 的样本。

$$T_i = \frac{a^i + a^{i+1}}{2}, \quad 1 \leq i \leq n-1$$



（有很多可能的划分点，取哪一个呢）

16.5 连续特征决策树

根据连续属性 a 和阈值 t ，对 D 进行划分，整个集合的信息熵变为：

$$Ent(D | a, t) = \frac{|D_t^+|}{|D|} Ent(D_t^+) + \frac{|D_t^-|}{|D|} Ent(D_t^-)$$

根据连续属性 a 和阈值 t ，对 D 进行划分所获得的信息增益：

$$Gain(D, a, t) = Ent(D) - Ent(D | a, t)$$

在特征选择的同时，确定合适的阈值 t （使得信息增益最大）

$$\max_{t=\{T_1, T_2, \dots, T_{n-1}\}} Gain(D, a, t)$$

编号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
密度	0.697	0.774	0.634	0.608	0.556	0.403	0.481	0.437	0.666	0.243	0.245	0.343	0.639	0.657	0.360	0.593	0.719
含糖率	0.460	0.376	0.264	0.318	0.215	0.237	0.149	0.211	0.091	0.267	0.057	0.099	0.161	0.198	0.370	0.042	0.103
敲声	浊响	沉闷	浊响	沉闷	浊响	浊响	浊响	浊响	沉闷	清脆	清脆	浊响	浊响	沉闷	浊响	浊响	沉闷
纹理	清晰	清晰	清晰	清晰	清晰	清晰	稍糊	清晰	稍糊	清晰	模糊	模糊	稍糊	稍糊	清晰	模糊	稍糊
脐部	凹陷	凹陷	凹陷	凹陷	凹陷	稍凹	稍凹	稍凹	稍凹	平坦	平坦	平坦	凹陷	凹陷	稍凹	平坦	稍凹
触感	硬滑	硬滑	硬滑	硬滑	硬滑	软粘	软粘	硬滑	硬滑	软粘	硬滑	软粘	硬滑	硬滑	软粘	硬滑	硬滑
好瓜	是	是	是	是	是	是	是	是	否	否	否	否	否	否	否	否	否

例子：如何根据表中的数据，选择根结点划分的属性？

密度：

{0.243,0.245,0.343,0.360,0.403,0.437,0.481,0.556,0.593,0.608,0.634,0.639,
0.657,0.666,0.697,0.719,0.774}

划分阈值：

{0.244,0.294,0.351,0.381,0.420,0.459,0.518,0.574,0.600,0.621,0.636,0.648,
0.661,0.681,0.708,0.746}

当 t 取0.244时：

$D_t^+ = \{1,2,3,4,5,6,7,8,9,11,12,13,14,15,16,17\}$, $D_t^- = \{10\}$

条件信息熵： $\text{Ent}(D|\text{密度},0.244) = 0.9412$

集合D的信息熵： $\text{Ent}(D) = 0.998$

信息增益： 0.0568

经过计算， t 取值0.381时，信息增益最大： $\text{Gain}(D, \text{密度})=0.262$

对于含糖率属性, t 取值0.126时, 信息增益最大

$$\text{Gain}(D, \text{含糖率})=0.349$$

$$\text{Gain}(D, \text{敲声})=0.141$$

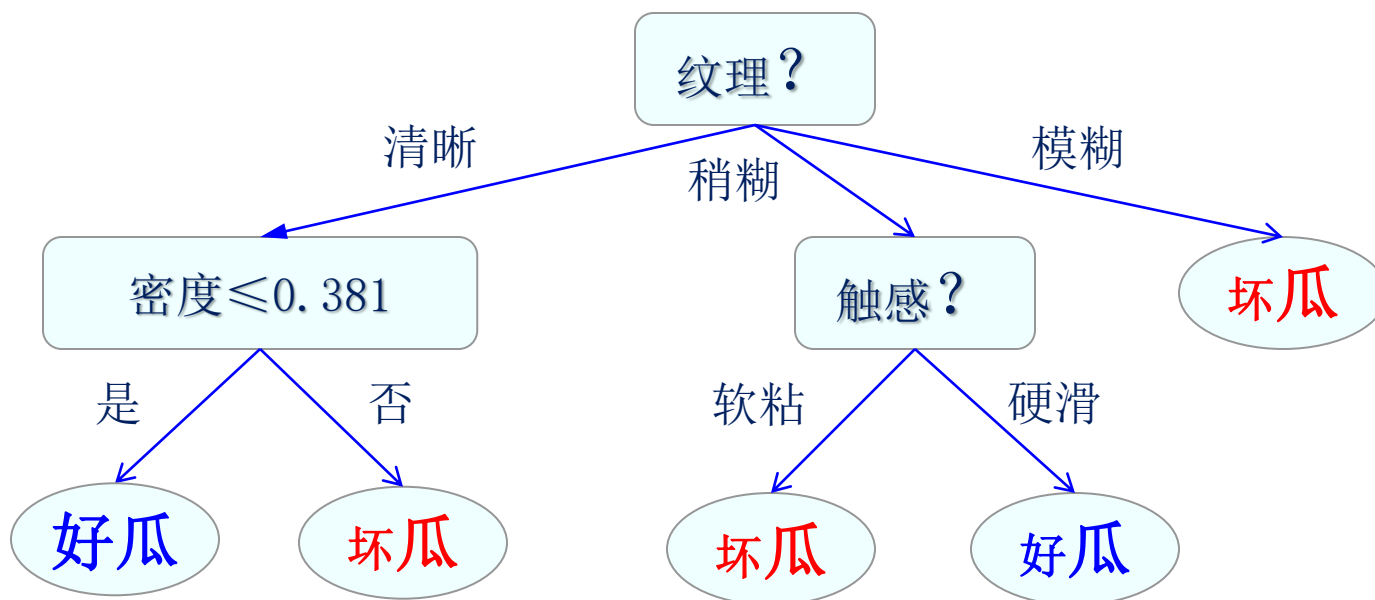
$$\text{Gain}(D, \text{纹理})=0.381$$

$$\text{Gain}(D, \text{脐部})=0.289$$

$$\text{Gain}(D, \text{触感})=0.006$$

$$\text{Gain}(D, \text{密度})=0.262$$

$$\text{Gain}(D, \text{含糖率})=0.349$$



16.6 CART

16.6 CART决策树

分类与回归树(classification and regression tree,CART)模型
由 Breiman 等人在1984年提出，是应用广泛的决策树学习方法。

- CART是在给定输入随机变量 X 条件下输出随机变量的条件概率分布的学习方法（有监督的学习）
- CART同样由特征选择、树的生成及剪枝组成，既可以用于分类也可以用于回归，以下将用于分类与回归的树统称为决策树
- CART是二叉树，内部结点特征的取值为“是”和“否”，左分支是取值为“是”的分支，右分支是取值为“否”的分支
- 该决策树等价于递归地二分每个特征，将输入空间（即特征空间）划分为有限个单元，并在这些单元上确定预测的概率分布，即在输入给定的条件下输出的条件概率分布

16.6 CART决策树

- CART是一个二叉树结构，内部结点取值“是”和“否”
- CART分类树使用基尼指数进行特征选择
- CART回归树使用平方误差最小准则进行特征选择
- CART决策树剪枝先根据数据自动产生一系列子树，然后通过交叉验证选择最优的子树作为剪枝结果

16.6.1 CART分类树—回顾基尼指数

基尼值与基尼指数：

属性 a 的基尼指数： 利用属性 a 对样本集合 D 划分后的基尼值之和。

$$Gini_index(D, a) = \sum_{i=1}^V \frac{|D^i|}{|D|} Gini(D^i)$$

特别地，如果样本集合 D 根据特征 a 是否取某一可能值 t 被分割成 D_1 和 D_2 两部分，即

$$D_1 = \{(\mathbf{x}, y) \in D | \mathbf{x}(a) = t\}, \quad D_2 = D - D_1$$

则在特征 a 的条件下，集合 D 的基尼指数定义为

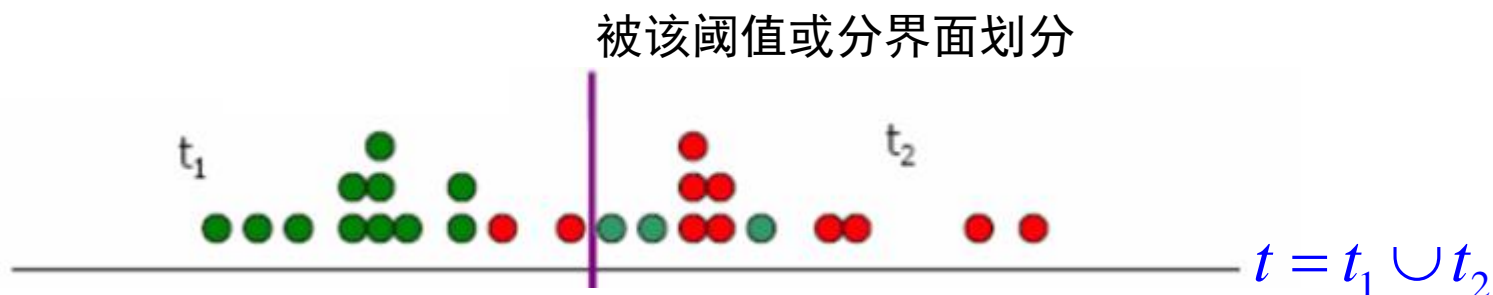
$$Gini_index(D, a) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2) \quad (\text{加权的基尼值})$$

- ✓ **基尼值 $Gini(D)$** 表示集合 D 的不确定性，基尼指数 $Gini_index(D, a)$ 表示经特征 a 取值 t 分割后集合 D 的不确定性。
- ✓ 基尼指数值越大，样本集合的不确定性也就越大，这一点与熵相似。

基尼指数计算示例：

$$Gini(t) = 1 - \sum_j p^2(j|t)$$

$$p(red|t) = 11/25, \quad p(green|t) = 14/25, \quad Gini(t) = 1 - 0.194 - 0.314 = 0.492$$



$$p(red|t_1) = 2/13$$

$$p(red|t_2) = 9/12$$

$$p(green|t_1) = 11/13$$

$$p(green|t_2) = 3/12$$

$$Gini(t_1) \approx 1 - 0.024 - 0.726 = 0.25$$

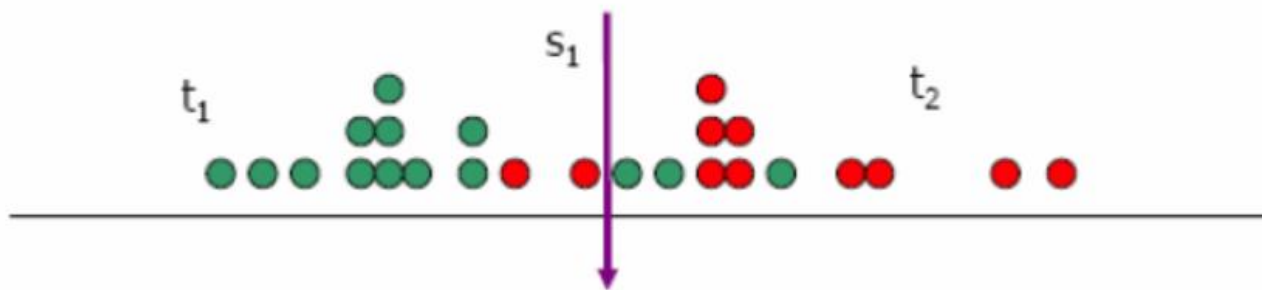
$$Gini(t_2) \approx 1 - 0.563 - 0.063 = 0.374$$

$$Gini_index = 0.25 * \left(\frac{13}{25}\right) + 0.374 * \left(\frac{12}{25}\right) = 0.31$$

16.6.1 CART分类树

划分的优劣：

基尼指数的变化量： $Gini(D) - Gini_index(D, a)$



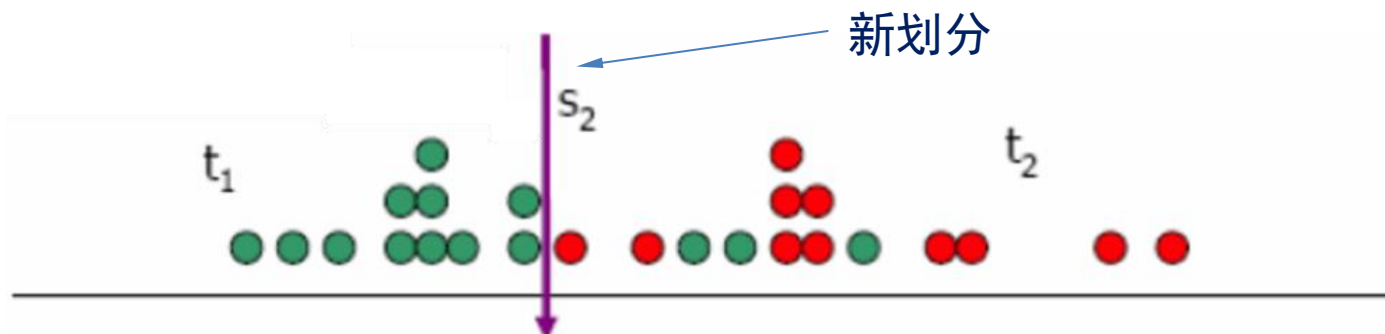
$$Improvement(s_1) = 0.492 - 0.31 = 0.182$$

$$\left(= gini(t) - \left(\frac{n_1}{n} \right) * gini(t_1) - \left(\frac{n_2}{n} \right) * gini(t_2) \right)$$

变化量越大，划分的纯度就越好！

$$Gini(t) = 1 - \sum_j p^2(j|t), \quad p(red|t) = 11/25, \quad p(green|t) = 14/25$$

$$Gini(t) = 1 - 0.194 - 0.314 = 0.492 \quad \leftarrow \text{数据混杂度}$$



$$p(red|t_1) = 0/11$$

$$p(red|t_2) = 11/14$$

$$p(green|t_1) = 11/11$$

$$p(green|t_2) = 3/14$$

$$Gini(t_1) = 1 - 0.0 - 1.0 = 0.0$$

$$Gini(t_2) = 1 - 0.617 - 0.046 = 0.337$$

$$Gini_index = 0.0 * \left(\frac{11}{25}\right) + 0.337 * \left(\frac{14}{25}\right) = 0.189$$

$$Improvement(s_2) = 0.492 - 0.189 = 0.303$$

s_2 是更好的划分

$$Improvement(s_1) = 0.492 - 0.31 = 0.182$$

16.6.1 CART分类树

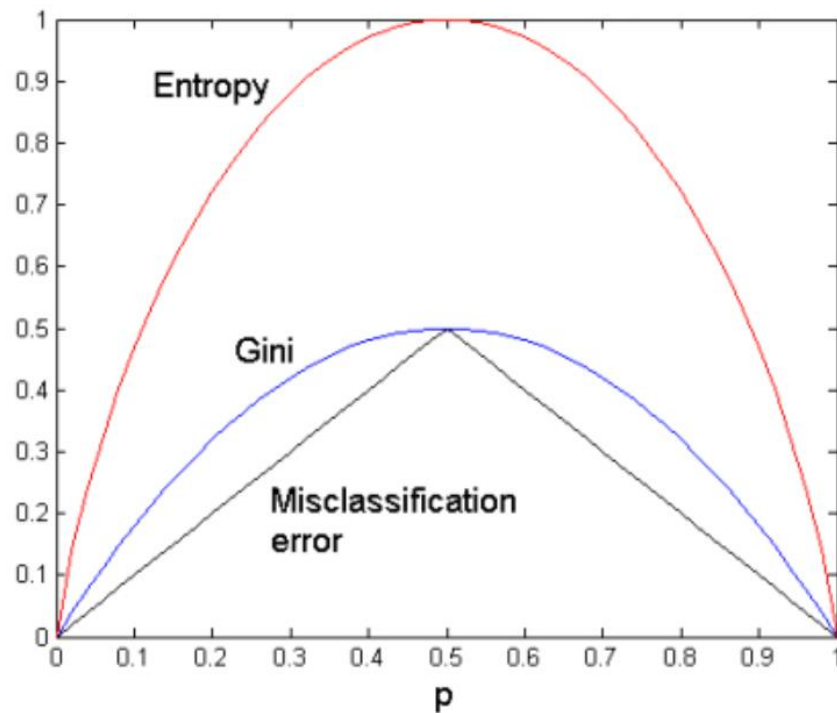
不同评价标准:

$$Gini(t) = 1 - \sum_j [p(j|t)]^2$$

$$Entropy(t) = - \sum_j p(j|t) \log p(j|t)$$

$$Error(t) = 1 - \max_i P(i|t)$$

For two class problems



16.6.1 CART分类树

输入: 训练数据集 D , 停止计算的条件;

输出: CART 决策树.

根据训练数据集, 从根结点开始, 递归地对每个结点进行以下操作, 构建二叉决策树:

1. 设结点的训练数据集为 D , 计算现有特征对该数据集的基尼指数。此时, 对每一个特征 A , 对其可能取的每个值 a , 根据样本点对 $A=a$ 的测试为“是”或“否”将 D 分割成 D_1 和 D_2 两部分, 计算 $A=a$ 时的基尼指数
2. 在所有可能的特征 A 以及它们所有可能的切分点 a 中, 选择基尼指数最小的特征及其对应的切分点作为最优特征与最优切分点. 依最优特征与最优切分点, 从现结点生成两个子结点, 将训练数据集依特征分配到两个子结点中去.
3. 对两个子结点递归地调用(1), (2), 直至满足停止条件.
4. 生成CART 决策树.

算法停止计算的条件是: (1) 结点中的样本个数小于预定阈值, (2) 样本集的基尼指数小于预定阈值 (样本基本属于同一类), 或者 (3) 没有更多特征.

16.6.2 CART回归树

决策树函数：一棵决策树对应了对输入空间的一种划分，每个划分的子区域对应一个决策值（决策类别）。

$$f(x) = \sum_{m=1}^M c_m I(x \in R_m)$$

R_m ：第 m 个子区域， c_m 是对应的决策值， $I()$ 指示函数。

c_m 对应子区域内所有样本的决策值，根据平方误差准则应满足：

$$c_m = \arg \min_c \sum_{x_i \in R_m} (y_i - c)^2$$

该子区域内所有样本y值的平均

16.6.2 CART回归树

CART回归树总体构建思路：

回归树构建：递归地选择最优的特征属性和其划分值，将当前数据空间划分为两部分，直到满足停止条件。

离散特征：通过**判断划分属性“是”和“不是”**进行划分

例如：色泽=青绿，色泽≠青绿

注：构建回归树时，也可能有离散特征

连续特征：根据与划分值 s 的**大小关系**进行划分

$$D^-(j, s) = \{x \mid x^j \leq s\}, \quad D^+(j, s) = \{x \mid x^j > s\}$$

停止条件：(1) 结点中样本个数小于预定阈值；(2) 没有可用特征；(3) 决策误差小于预定阈值。

16.6.2 CART回归树

特征选择（如何选择最优属性）：使用平方误差最小准则

$$\arg \min_{j,s} \left[\sum_{x_i \in D^+(j,s)} (y_i - m^+)^2 + \sum_{x_i \in D^-(j,s)} (y_i - m^-)^2 \right]$$

m^+ ： $D^+(j, s)$ 内所有样本 y 值的均值

m^- ： $D^-(j, s)$ 内所有样本 y 值的均值

方法：**遍历**所有特征 j 和 可能的划分值 s ，选择使得预测平方误差最小的组合，同时**记录对应的均值**，作为决策函数相应区域的取值。

16.6.2 CART回归树

最小二乘回归树生成算法

输入:训练数据集D;

输出:回归树 $f(x)$.

在训练数据集所在的输入空间中, 递归地将每个区域划分为两个子区域并决定每个子区域上的输出值, 构建二叉决策树;

(1) 选择最优切分变量 j 与切分点 s , 求解

$$\min_{j,s} [\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2]$$

遍历变量 j , 对固定的切分变量 j 扫描切分点 s , 选择使上式达到最小值的对 (j, s) .

(2) 用选定的对 (j, s) 划分区域并决定相应的输出值:

$$R_1(j, s) = \{x | x^{(j)} \leq s\}, \quad R_2(j, s) = \{x | x^{(j)} > s\}$$

$$\widehat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m(j,s)} y_i, \quad x \in R_m, \quad m = 1, 2$$

(3)继续对两个子区域调用步骤(1), (2), 直至满足停止条件.

(4)将输入空间划分为 M 个区域 R_1, R_2, \dots, R_M , 生成决策树: $f(x) = \sum_{m=1}^M \widehat{c}_m I(x \in R_m)$

假设 X 与 Y 分别为输入和输出变量, 并且 Y 是连续变量, 给定训练数据集

$$D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$$

考虑如何生成回归树.

16.6.3 CART剪枝

CART剪枝的基本思路

1. 自底向上进行剪枝，产生一系列剪枝后的子树；
2. 通过交叉验证，在独立的验证数据集上对所有子树进行测试，选择最优一个子树。

核心问题：如何产生一系列剪枝后的子树？

16.6.3 CART剪枝

CART剪枝的目标：决策树整体代价最小。

采用下式来描述模型预测错误的程度，越小越好：

$$C_{\alpha}(T) = C(T) + \alpha|T| \quad (\text{是 } T \text{ 和 } \alpha \text{ 的函数})$$

- T 表示一个决策树
- $C(T)$ 是对训练数据的预测误差，比如分类用基尼指数表示，回归用均方误差表示
- $|T|$ 表示树 T 的叶节点个数。

- ✓ α 从小变到大，决策树越来越精简（叶结点越来越少）；
通过设置一系列 α 值，产生一系列剪枝后的子树。
- ✓ α 越大，模型趋于少分枝。 α 越小，模型越趋于多分枝。

16.6.3 CART剪枝

考虑某个内部结点 t （不是叶节点），剪枝前的整体代价：

$$C_B = C(T') + C(T_t) + \alpha|T'| + \alpha|T_t|$$

（通过包含与不包含由节点 t 为根节点的子树来理解）

剪枝后的整体代价：

$$C_A = C(T') + C(t) + \alpha|T'| + \alpha$$

（整棵树变成： $T' + t$ ，且 t 为叶子节点）

剪枝后代价应该**不高于**剪枝前： $\alpha \geq \frac{C(t) - C(T_t)}{|T_t| - 1} = g(t)$

（注：对于任意结点 t ，记以 t 为根节点的子树为 T_t ，只有 t 一个结点的树直接记为 t ）

16.6.3 CART剪枝

CART剪枝算法 (序列输出)

输入：CART生成的决策树 T_0

输出：一系列剪枝后的子树 T_0, T_1, \dots, T_n

1、令 $k=0$, $T=T_0$, $\alpha=+\infty$;

2、自下而上地对 T 各内部结点 t 计算：

$$g(t) = \frac{C(t) - C(T_t)}{|T_t| - 1} \quad \underline{\alpha = \min(\alpha, g(t))}$$

α 取当前最小：控制每次只剪掉一个结点的枝

其中， T_t 表示以 t 为根结点的子树， $C(T)$ 表示利用树 T 对训练数据的预测误差。

3、对所有结点中 $g(t)$ 值最小的那个进行剪枝，叶结点 t 的类别为其中多数样本的类别，得到树 T ，记录 $\alpha=g(t)$, $k=k+1$, $T_k=T$;

4、如果 T 不是由根结点及两个叶结点构成的树，返回2，否则结束；

16.6.3 CART剪枝

CART剪枝算法

1. 决策树生成： 基于训练数据集生成决策树，生成的决策树要尽量大；
2. 决策树剪枝： 用验证数据集对已生成的树进行剪枝并选择最优子树，这时用损失函数最小作为剪枝的标准.

决策树的生成就是递归地构建二叉决策树的过程. 对回归树用平方误差最小化准则，对分类树用基尼指数(Gini index)最小化准则，进行特征选择，生成二叉树.

16.6.3 CART剪枝

在终止条件被满足，划分停止之后，下一步是剪枝：

- 给树剪枝就是剪掉“弱枝”，弱枝指的是在验证数据上误分类率高的树枝
- 对树剪枝会增加训练数据上的错误分类率，但精简的树会提高新记录上的预测能力
- 剪掉的是最没有预测能力的枝

16.7 决策树小结

16.7.1 小结

决策树模型：以树形结构，对输入数据的不同属性进行判定，根据多次判定结果决定输入数据类别

- 是if-then规则的集合，可解释性好
- 树形结构，分而治之，分类速度快

决策树学习：根据带类别标签的训练数据，自动地构建决策树的过程。

- 决策特征选择
- 决策树的生成
- 决策树的剪枝

16.7.1 小结

- 决策特征选择准则：特征选择的目的是为了为了使决策树的分支结点包含的样本尽可能属于同一类。
- **信息增益**：给定某个特征的前提下，数据集信息熵减少的程度（确定性提高的程度），对取值个数较多的离散特征有偏向。
- **信息增益率**：利用特征属性的“固有不确定性”对信息增益进行归一化，解决偏向取值个数较多的特征问题。
- **基尼指数**：给定某个特征的前提下，数据集在不同特征取值下的纯度。

16.7.1 小结

- 决策树生成：递归地进行决策特征选择，构建树的结点的过程。
- ID3算法：利用信息增益最大的准则进行特征选择。
- C4.5算法：结合信息增益，利用增益率最大的准则进行特征选择。
- CART分类树：利用基尼指数最小的准则进行特征选择。
- CART回归树：利用最小平方误差的准则进行特征选择。

16.7.1 小结

决策树剪枝：修剪一些不必要的分支，**减少决策树复杂度，提升泛化能力（或减少整体损失），减少过拟合。**

- **预剪枝**：在决策树生成的过程进行剪枝，只有泛化性能提升的结点划分才被采纳。**计算速度快，有欠拟合的风险。**
- **后剪枝**：先生成决策树，再自底向上对所有结点进行检验，判断剪枝前后是否会带来泛化性能的提升，以此决定是否剪枝。**欠拟合风险低，泛化能力强，计算量大。**

16.7 小结

处理连续特征取值：

- 设置阈值，**将连续特征取值划分为两个子集合**，转化为取值个数为2的离散情况。

处理缺失特征：

- 选择划分属性时，**只使用没有缺失特征的数据子集**计算特征选择标准，利用无缺失数据的比例进行加权
- 对于划分属性上没有取值的样本，按照当前子结点的大小比例**划分至所有子结点**，更新样本的权重

16.7.1 小结

CART决策树：以**二叉树**形式进行组织的决策树，分CART回归决策树和CART分类决策树。

- **回归树**：使用**最小平方误差**准则进行特征选择。
- **分类树**：使用**基尼指数最小**准则进行特征选择。
- **剪枝**：主要采用后剪枝策略，同时生成一系列剪枝结果，通过**交叉验证**确定最优的剪枝结果。

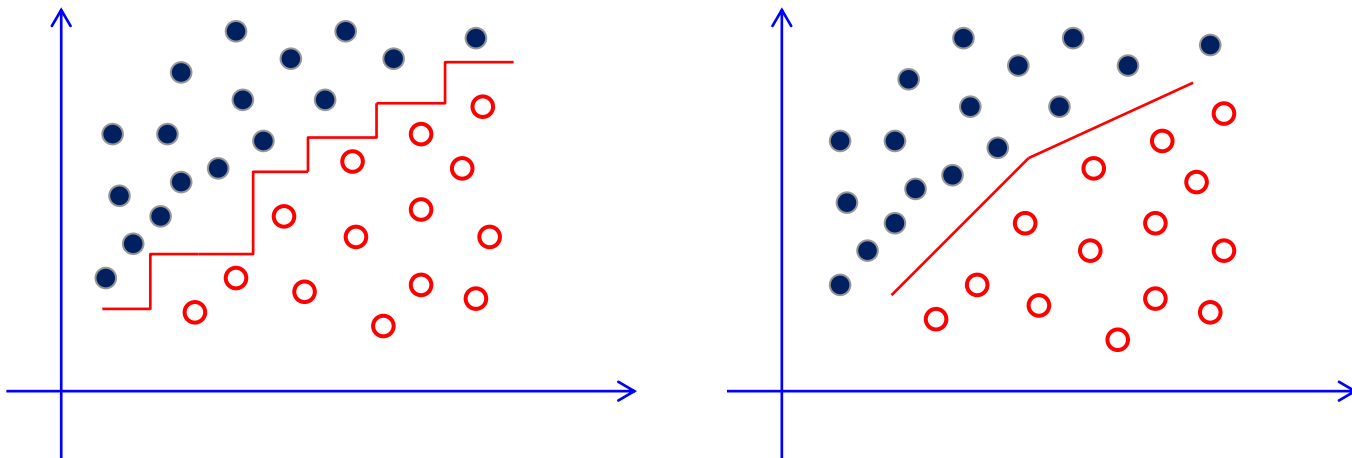
16.7.2 决策树推广

- 多变量决策树
- 随机森林

16.7.2 决策树推广

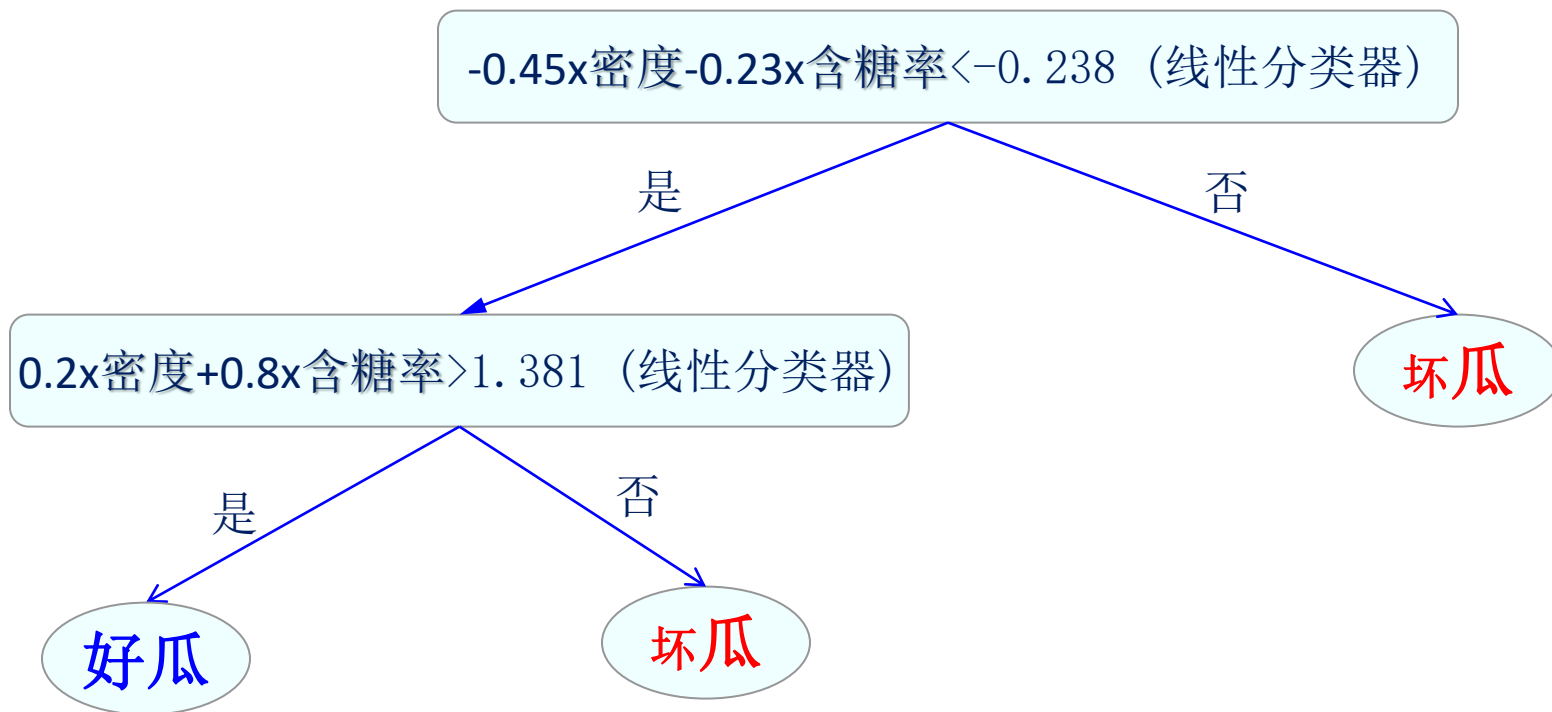
单变量决策树：决策树每个节点根据**一个特征属性**的取值进行分支。决策函数是这些分支的组合，其决策面表现为**轴平行的特性**，即，由若干个和坐标轴平行的线段组成。

多变量决策树：决策树每个结点根据若干个特征属性的线性组合进行分支，**每个节点对应一个线性分类器**。决策函数是这些线性分类器的组合，其决策面表现为分段线性。



16.7.2 决策树推广

多变量决策树



16.7.2 决策树推广

随机森林：通过**随机方法构建多个决策树**，利用决策树分类结果进行投票，得到随机森林的分类结果。随机森林泛化能力更强。

随机方法：生成决策树之前，先采用随机方法确定可用的**训练数据**和**特征属性**。

- **训练数据随机**：随机采样一部分训练数据
- **使用特征随机**：随机选择一部分特征属性

16.8 Random Forests (随机森林)



16.8.1 随机森林的思想来源

主要作者:

- **Leo Breiman** was a distinguished **statistician** at the University of California, Berkeley.
 - Breiman's work helped to bridge the gap between statistics and computer science, particularly in the field of **machine learning**.
 - His most important contributions were his work on **classification and regression trees (CART)** and **ensembles of trees fit to bootstrap samples**.
 - Bootstrap aggregation was given the name **bagging** by Breiman.
 - Another of Breiman's ensemble approaches is the **random forest**.

16.8.1 随机森林的思想来源

随机森林的通俗理解:

- Random Forest, 顾名思义, Random就是随机抽取, Forest是指不止一棵树, 而由一群决策树组成的一片森林。
 - 用随机抽取的方法训练出一群决策树来完成分类任务。
- 为了解决样本数量有限的问题, RF用了两次随机抽取:
 - 训练样本的随机抽取 + 对变量（特征）的随机抽取。
- RF的核心是由弱变强思想的运用。
 - 每棵决策树由于只用了部分变量、部分样本训练而成, 可能单个的分类准确率并不是很高。
 - 但是当一群这样的决策树组合起来分别对输入数据作出判断时, 可以带来较高的准确率。有点类似于俗语 三个臭皮匠顶个诸葛亮。
- 随机森林有两个重要参数:
 - 一是树节点预选的变量个数;
 - 二是随机森林中树的个数

16.8.1 随机森林的思想来源

在说明random forest的算法之前，先了解一下它的思想来源，主线条可以由下面这个发展线来表示。

PAC → Bootstrap → Bagging → Random Forest ← CART

16.8.1 随机森林的思想来源

来源1: PAC (Probably Approximately Correct)

- Kearns和Valiant提出:
 - 若存在一个多项式级的学习算法来识别一组概念，并且识别正确率很高，那么这组概念是**强可学习**算法；
 - 如果学习算法识别一组概念的正确率仅比随机猜测略好，那么这组概念是**弱可学习**算法。
 - 如果可以将**弱学习算法提升成强学习算法**，那么就只要找到一个弱学习算法，然后把它提升成强学习算法，而不必去找通常情况下很难获得的强学习算法。

16.8.1 随机森林的思想来源

来源2: Bootstrap (用于基本统计的估计)

- 根据PAC由弱得到强的思想，统计学著名学者Bradley Efron在1979年提出了Bootstraps算法，这个名字来自于成语“pull up by your own bootstraps”，意思是依靠自己的资源，称为自助法。
 - 核心思想：当样本数量不大，分布未知时，可以从原始样本中随机抽取的多个样本（弱学习）来估计原样本真实的分布情况。
（是非参数统计中一种重要的统计量方差估计方法，在此基础上进行区间估计的统计方法。）
 - 比如方差估计，其基本步骤如下：
 - ① 从原始数据集中，有放回地抽样一定数量的样本
 - ② 根据抽出的样本计算给定的统计量 T
 - ③ 重复上述 N 次（一般大于1000），得到 N 个统计量 T
 - ④ 计算上述 N 个统计量 T 的样本方差，得到统计量的方差

16.8.1 随机森林的思想来源

来源2: Bootstrap (续)

- 这里举例说明其中一种最常用的方法：**0.632自助法**。
 - 假设给定的数据集包含 d 个样本。该数据集有放回地抽样 d 次，产生 d 个样本的训练集。
 - 原数据中的某些样本很可能在该样本集中出现多次
 - 显然每个样本被选中的概率是 $1/d$
 - 因此，未被选中的概率就是 $(1-1/d)$
 - 这样，一个样本在训练集中没出现的概率就是 d 次都未被选中的概率，即 $(1-1/d)^d$ 。当 d 趋于无穷大时，这一概率就将趋近于 $e^{-1}=0.368$
 - 所以，留在训练集中的样本大概就占原来数据集的63.2%。

16.8.1 随机森林的思想来源

来源3: Bagging（基于样本采样的多分类器集成）

- Bagging又叫**bootstrap aggregation**，是Breiman在1993年提出的方法，第一步就是根据Bootstrap进行抽样。
- 基本的步骤：
 - ① 从样本集中用Bootstrap采样选出 n 个样本
 - ② 在所有属性上，对这 n 个样本建立分类器（CART or SVM or ...）
 - ③ 重复以上两步 m 次，i.e.建立 m 个分类器（CART or SVM or ...）
 - ④ 将数据放在这 m 个分类器上跑，最后vote看到底分到哪一类
- 该方法可以大大降低每个分类器的不稳定性，从而带来较高的预测准确率。
- 从该方法再往下发展就是随机森林了。

16.8.1 随机森林的思想来源

From Bagging to Random Forest:

- 随机森林是以决策树为基本分类器的一个集成学习模型，它包含多个由Bagging集成学习技术训练得到的决策树。
- Random Forests不同的是：在Bagging的基础上，使用一种改进的树学习算法，在每个候选分裂的学习过程中，选择特征值的一个随机子集。有时被称为“feature bagging”。

16.8.1 随机森林的思想来源

来源4: CART——回顾其构造过程

回顾一下RF里面将用到的决策树算法。

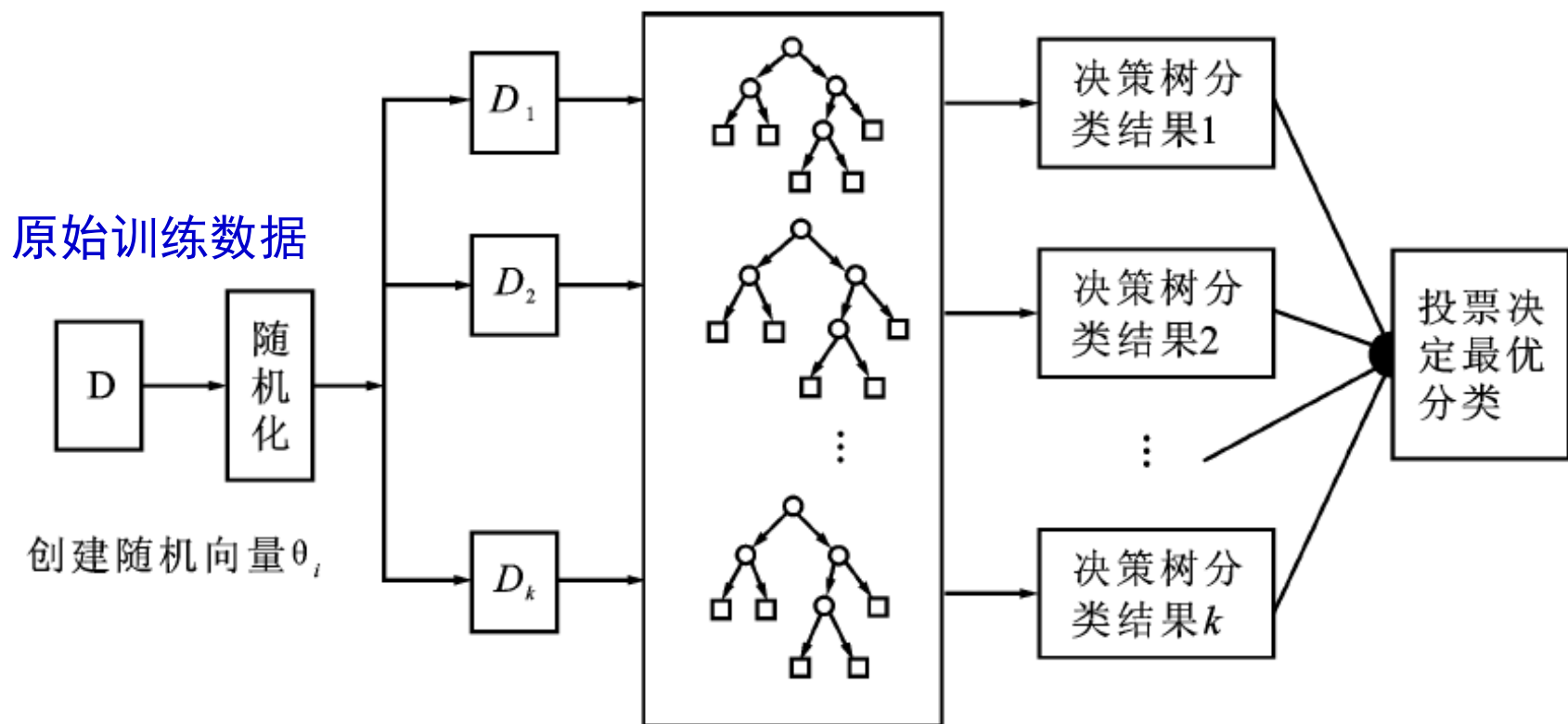
CART是以自顶向下递归方式构造的二叉树。基本的步骤包括构造和剪枝两部分。

– 比如：构造决策树

- ① 创建一个结点N
- ② 如样本T都在同一个类C中，返回N作为叶结点，以类C做标记
- ③ 从候选属性A集中找出GINI系数最小的属性
- ④ 将样本T划分为T1， T2两个子集
- ⑤ 对T1重复①-④
- ⑥ 对T2重复①-④
- 生成的树是完全分裂的，存在过度拟合的现象，CART使用事后剪枝的方式对决策树进行剪枝。

16.8.2 随机森林算法

建立多棵决策树



随机森林分类过程图

16.8.2 随机森林算法

- 该算法用随机的方式建立起多棵决策树，然后由这些决策树组成一个森林，其中每棵决策树之间没有关联，当有一个新的样本输入时，就让每棵树独立的做出判断，按照多数原则决定该样本的分类结果。
- 基本任务：
 - 构建随机森林
 - 使用随机森林预测

构建随机森林:

- ① 从样本集中用bagging采样选出 n 个样本，预建立CART
- ② 在树的每个节点上，从所有属性中随机选择 k 个属性，选择一个最佳分割属性作为节点（RI 和 RC）
- ③ 重复以上两步 m 次，i.e. 构建 m 棵CART(不剪枝)
- ④ 这 m 个CART形成Random Forest

与bagging方法不同，随机森林存在两次随机过程：

- 使用Bootstrap随机选择子样本；
- 最佳分割属性不是从完全的属性集上挑选出来，而是从随机选出的部分属性集中挑选的。

对于样本采样，Breiman采用0.632自助法。

对于属性的采样，Breiman设计了两种方法，分别是**RI(随机输入选择)**和**RC（随机线性组合）**：

- RI就是随机地从完全属性集中选择一定数量的属性形成候选属性集。
- RC会先从所有随机变量里面选择 L 个变量，然后把这些变量通过线性组合形成一个新的组合变量，使用的系数来自于 $[-1,1]$ 上的随机数。用这种方法得到 F 个组合变量形成候选分割属性集。

利用随机森林预测:

基本步骤（分类）：

- ① 向建立好的随机森林中输入一个新样本
- ② 随机森林中的每棵决策树都独立的做出判断
- ③ 将**得到票数最多**的分类结果作为该样本最终的类别

上面是针对分类而言的，对于回归预测，最简单的做法就是将所有决策树的**预测结果取平均值**作为最终的结果。

16.8.3 讨论

影响随机森林分类性能的主要因素

- 森林中单颗树的**分类强度**（Strength）：每颗树的分类强度越大，则随机森林的分类性能越好。
- 森林中树之间的**相关度**（Correlation）：**树之间的相关度越大**，则随机森林的分类性能越差。

16.8.3 讨论

随机森林的几个理论要点：

- 收敛定理 (详见相关论文，略)
 - 它度量了随机森林对给定样本集的分类错误率
- 泛化误差界
 - 单个决策树的分类强度越大，相关性越小，则泛化误差界越小，随机森林分类准确度越高。
- 袋外估计
 - Breiman在论文中指出袋外估计是无偏估计，袋外估计与用同训练集一样大小的测试集进行估计的精度是一样的。
 - Using out-of-bag estimates to monitor error, strength, and correlation

16.8.3 讨论

随机森林的优点

- ① 可以处理高维数据，可以不用进行特征筛选
- ② 非常容易进行分布式处理，实现起来非常高效
- ③ 可以利用OOB(out of bag)数据评价算法的误差率，而不用像一般算法那样还需要选择validation set测误差
- ④ 因为最终结果通过voting by majority得到，相对光滑，像SVM那样“large-margin like”的边界，受噪音干扰小，鲁棒性好
- ⑤ 森林中的树采用CART的优点：容易理解，处理非线性的问题，可以很好的处理数值型和序列型变量，处理缺失值

致谢：本课件是在张煦尧研究员和樊彬教授的PPT的基础上
综合和部分加工而成

致谢：本材料参考了多种来源，其中主要包含了周志华老师著作：
周志华 著：机器学习, 清华大学出版社, 2015.

Thank All of You!
(Questions?)

向世明

smxiang@nlpr.ia.ac.cn

peopleucas.ac.cn/~xiangshiming

时空数据分析与学习课题组(STDAL)

中科院自动化研究所·模式识别国家重点实验室