中山大学软件工程学院

SCHOOL OF SOFTWARE ENGINEERING

决策树

廖国成

liaogch6@mail.sysu.edu.cn

内容

- 基本流程
- 特征选择
- 剪枝

背景介绍

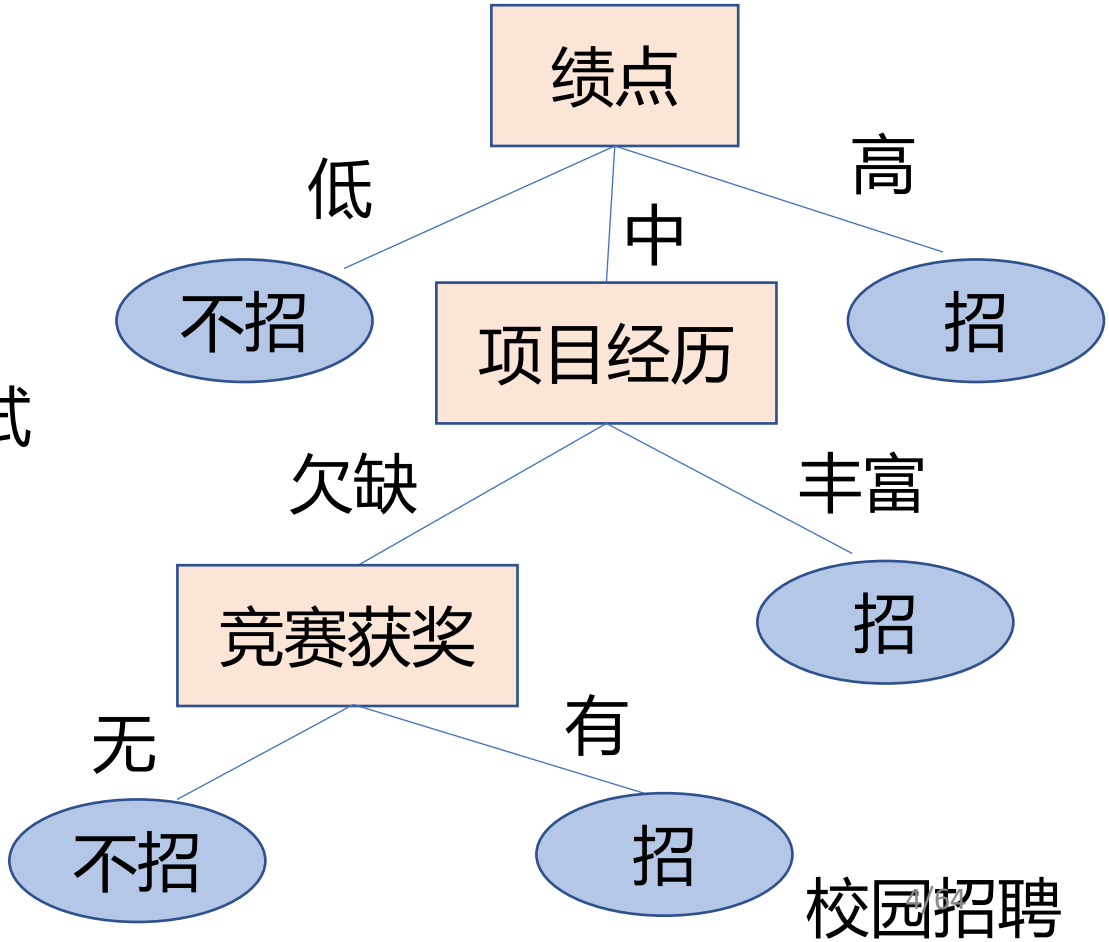
- 决定是否要去A餐厅吃饭
  - 距离远不远?
  - 价格贵不贵?
  - 好不好吃?
- 决定周末是否出去聚会
  - 有没有时间?
  - 聚会主题是否感兴趣?
  - 远不远?

在做决策的时候，基于多个因素，一步一步地做出判断 → 决策树

决策树定义

决策树是一种基本的分类方法，决策树模型呈树形结构，主要由结点（根结点、内部结点和叶结点）和边组成。

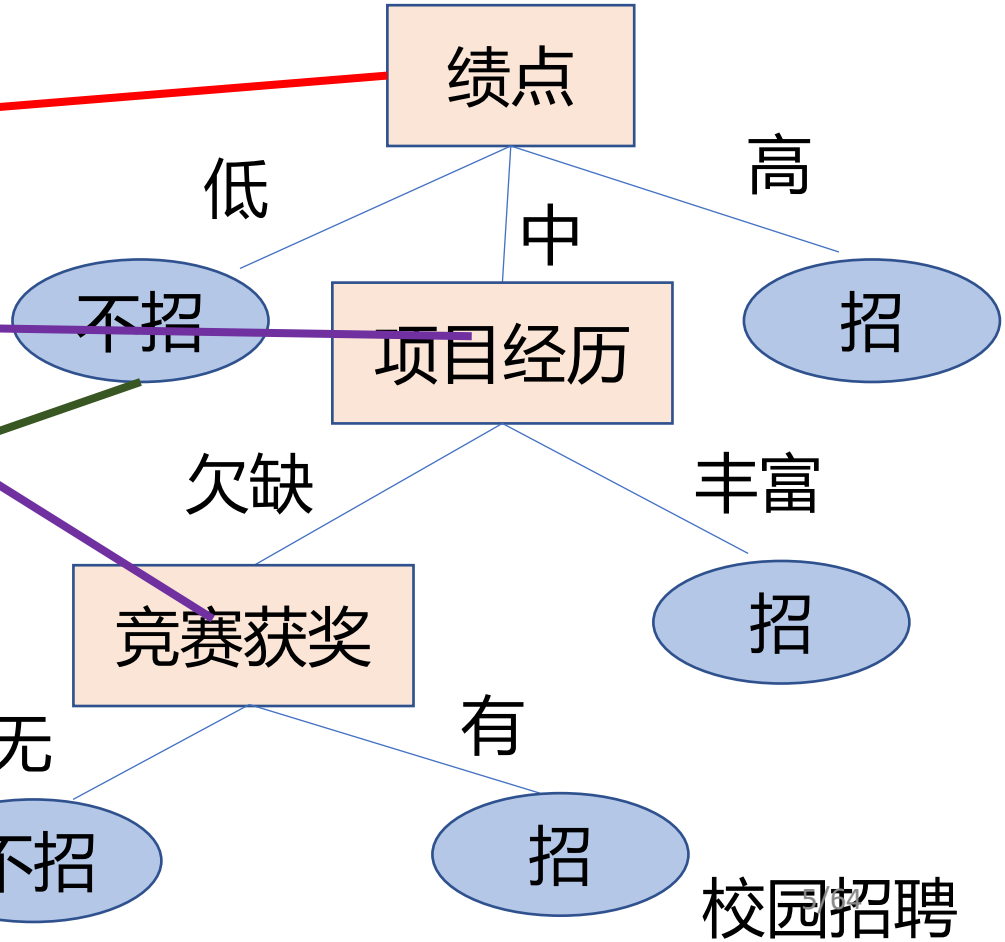
- 根结点（一个）
  包含样本全集
- 内部结点（若干）
  对应于一个属性的测试
- 叶结点（若干）
  对应于分类结果



决策树定义

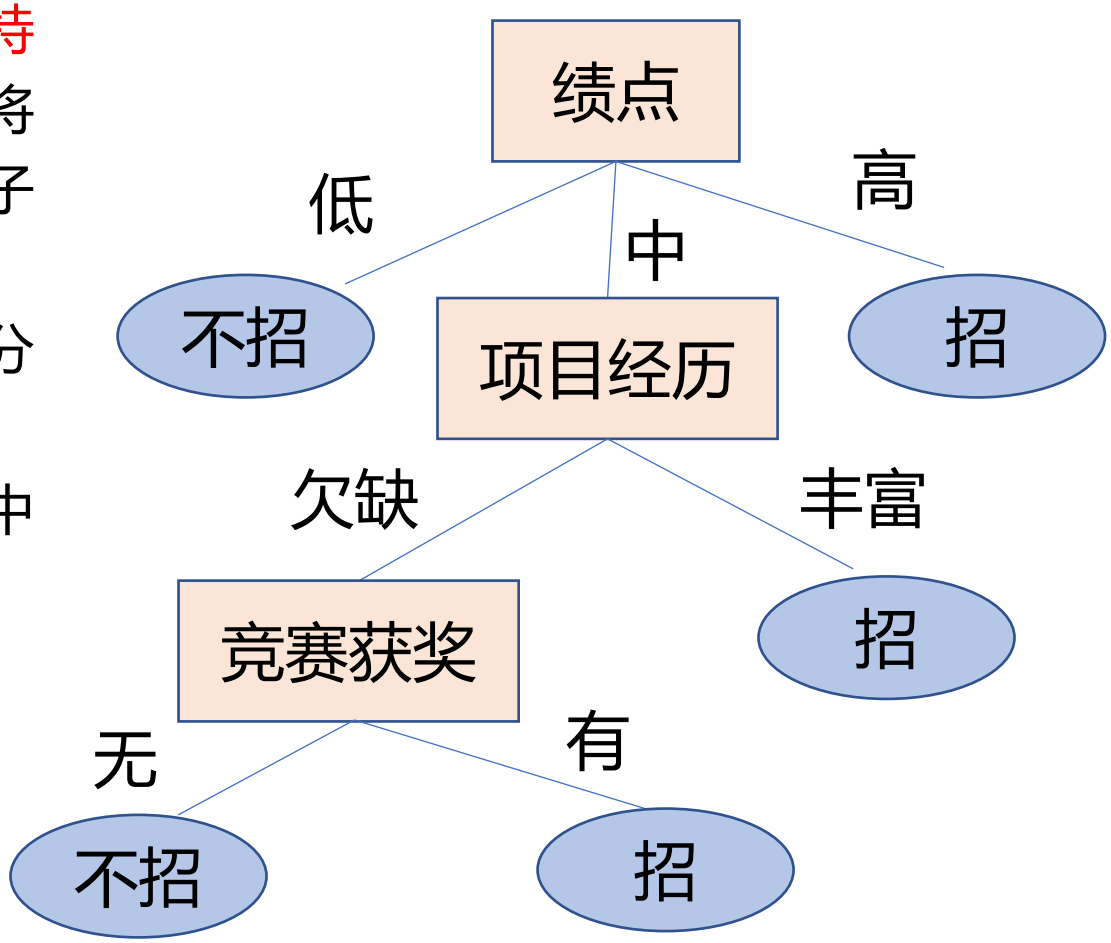
决策树是一种基本的分类方法，决策树模型呈树形结构，主要由结点（根结点、内部结点和叶结点）和边组成。

- 根结点（一个）
  包含样本全集
- 内部结点（若干）
  对应于一个属性的测试
- 叶结点（若干）
  对应于分类结果



决策树流程

- 从根结点开始，对示例的某一特征进行测试，根据测试结果，将示例分配到其子结点；每一个子结点对应着该特征的一个取值
- 如此递归地对实例进行测试并分配，直至达到叶结点
- 最后将示例分配到叶结点的类中



校园招聘

优缺点

- 优点
  - 可解释性强（商业数据、医疗数据）
  - 简单，易于实现
  - 能处理离散型和连续型数据
- 缺点
  - 容易过拟合
  - 对数据噪声敏感

随堂小测

决策树中的内部结点代表什么（）

A. 决策结果

B. 属性测试条件

C. 数据样本

D. 分类标签

决策树中的叶结点代表什么（）

A. 属性测试条件

B. 决策结果或分类标签

C. 数据样本

D. 分支条件

随堂小测

以下哪项是决策树的优点（）

A. 模型可解释性差

B. 只能处理数值型数据

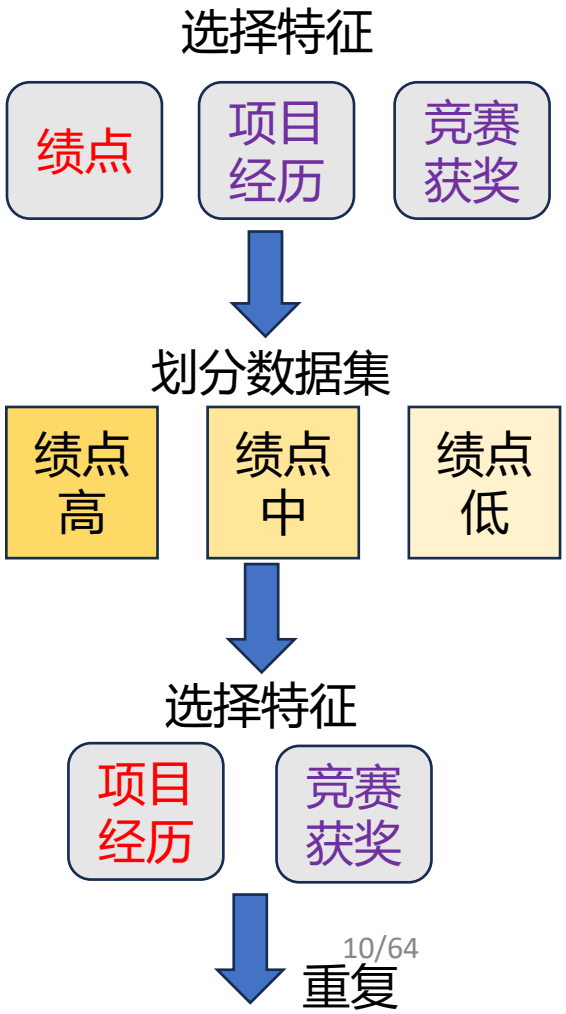
C. 对数据中的噪声不敏感

D. 易于理解与可视化

决策树的构建

从根结点开始，通过选择最优特征逐步划分数据集，直到满足停止条件（如达到最大深度、节点纯度足够高或数据集不能再分）

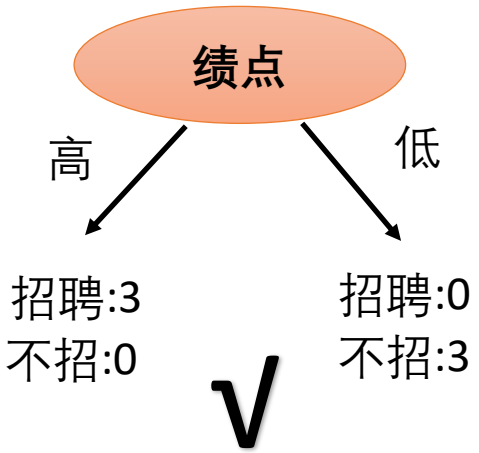
1. 选择最优特征：从当前数据集中选择一个特征，使得数据集的划分效果最好
2. 划分数据集：根据选择的特征，将数据集划分为多个子集
3. 递归构建子树：对每个子集重复上述过程，直到满足停止条件
4. 生成决策树：最终得到树形结构

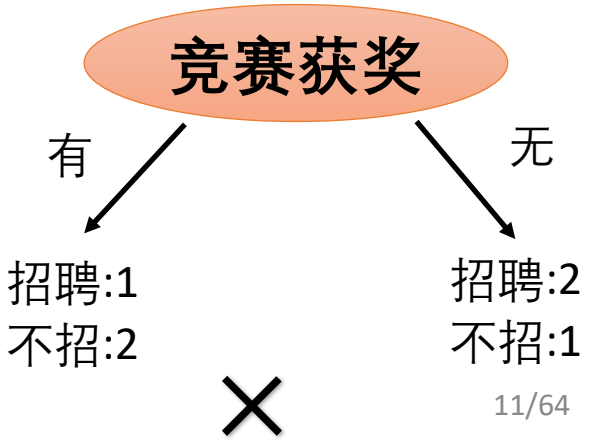


选择准则

如何选择最优划分特征：希望决策树的分支结点所包含的样本尽可能属于同一类别，即结点的“纯度” 越高越好

绩点	竞赛获奖	招聘与否
高	有	1
高	无	1
高	有	1
低	无	0
低	无	0
低	有	0



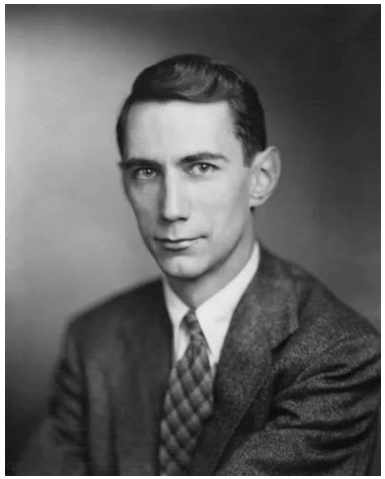


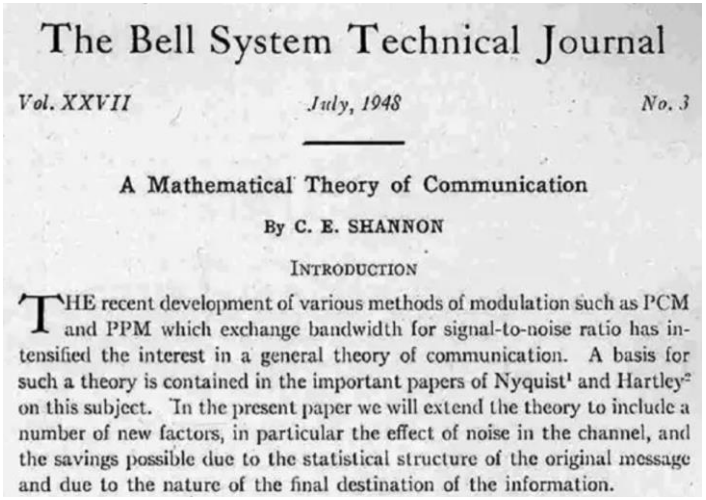
选择标准

- 信息增益
- 信息增益率
- 基尼系数

信息熵

- 什么是信息量？如何衡量信息量的大小
- 1948年香农在他著名的论文“通信的数学原理”中提出了信息熵，信息熵描述了信息的不确定性





The Bell System Technical Journal

Vol. XXVII *July, 1948* *No. 3*

A Mathematical Theory of Communication

By C. E. SHANNON

INTRODUCTION

THE recent development of various methods of modulation such as PCM and PPM which exchange bandwidth for signal-to-noise ratio has intensified the interest in a general theory of communication. A basis for such a theory is contained in the important papers of Nyquist[1] and Hartley[2] on this subject. In the present paper we will extend the theory to include a number of new factors, in particular the effect of noise in the channel, and the savings possible due to the statistical structure of the original message and due to the nature of the final destination of the information.

信息熵

- 假定当前样本集合D中第k类样本所占的比例为p_k，则D的信息熵 (entropy)定义为

$$\mathrm{Ent}(D) = -\sum_{k=1}^{K} p_k \log_2 p_k$$

- 计算信息熵时约定：若$p_k = 0$，则$p_k \log_2 p_k = 0$
- $\mathrm{Ent}(D)$的最小值为0
- 信息熵越大，不确定性越大

信息熵

假设有一个数据集分为正例和反例，考虑以下三种情况

- （1）20%正例，80%反例

$$-\frac{2}{10}\log_2\frac{2}{10}-\frac{8}{10}\log_2\frac{8}{10}=0.722$$

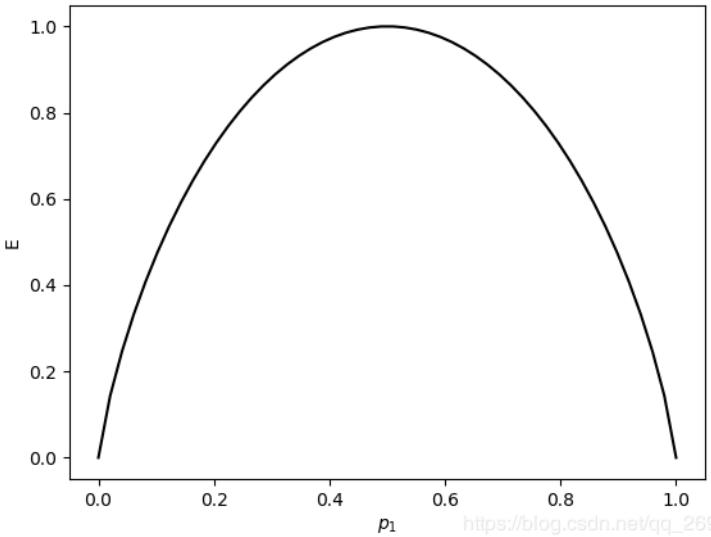
- （2）50%正例，50%反例

$$-\frac{5}{10}\log_2\frac{5}{10}-\frac{5}{10}\log_2\frac{5}{10}=1$$

- （3）80%正例，20%反例

$$-\frac{8}{10}\log_2\frac{8}{10}-\frac{2}{10}\log_2\frac{2}{10}=0.722$$

$$\mathrm{Ent}(D)=-\sum_{k=1}^{K}p_k\log_2 p_k$$



$$\mathrm{Ent}=-p_1\log_2 p_1-(1-p_1)\log_2(1-p_1)$$

信息熵越大，不确定性越大

随堂小测

以下情况中，（）信息熵最大，（）确定性最大

A. 20%正例，80%反例

B. 30%正例，70%反例

C. 40%正例，60%反例

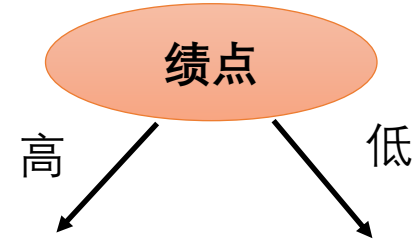
D. 50%正例，50%反例

信息增益

- 信息增益=当前数据集的信息熵-属性划分后的信息熵
- 信息增益描述了通过这个特征的划分，数据集的不确定性降低了多少

绩点	招聘与否
高	1
高	1
高	1
低	0
低	0
低	0

划分前
信息熵为1



绩点	招聘与否
高	1
高	1
高	1

绩点	招聘与否
低	0
低	0
低	0

划分后信息熵为0

数据集的信息熵

- 信息增益=当前数据集的信息熵-属性划分后的信息熵

校园招聘：100个学生数据，其中60人为“招”，

40人为“不招”

该数据集包含100个训练样本，类别2个，其中招的占$p_1 = 0.6$，不招的占$p_2 = 0.4$

$$\begin{aligned}\mathrm{Ent}(D) &= -\sum_{k=1}^{2} p_k \log_2 p_k \\ &= -(0.6 \times \log_2 0.6 + 0.4 \times \log_2 0.4) \\ &= 0.971\end{aligned}$$

属性划分后的信息熵

- 信息增益=当前数据集的信息熵-属性划分后的信息熵

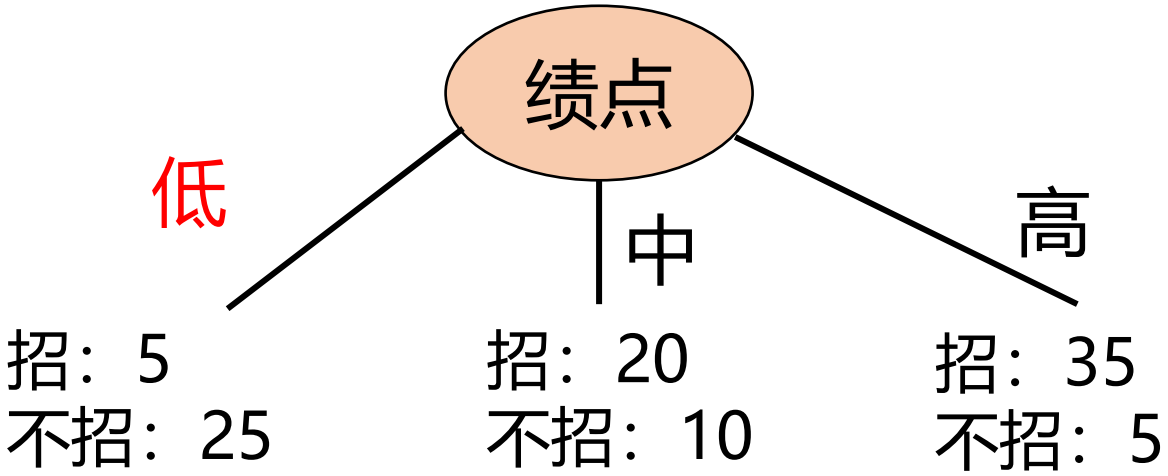
离散属性a有N个可能的取值$\{a^1, a^2, \cdots, a^N\}$，
属性$a = a^n$的样本集合，记为D^n。

1. 计算数据集D^n的信息熵$\mathrm{Ent}(D^n)$
2. 以子集的样本数量占总样本数的比例为权重，
   计算加权求和后的信息熵：

$$\sum_{n=1}^{N} \frac{|D^n|}{|D|} \mathrm{Ent}(D^n)$$

举例

1. 计算各个子集的信息熵；2. 加权求和



$\mathrm{Ent}(D^1) = 0.752$

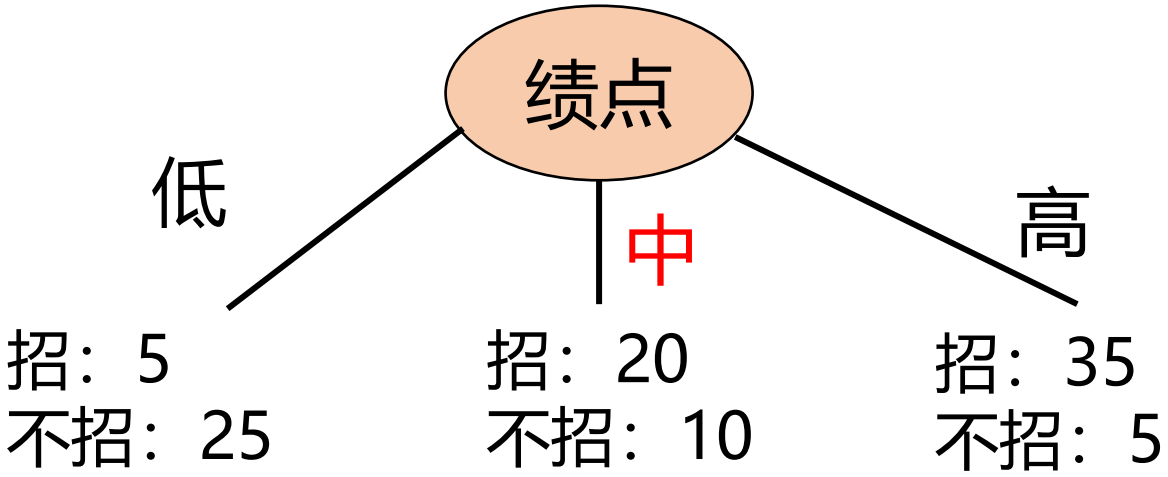
绩点=低的数据集包含30个训练样本，

招的占$p_1 = \frac{5}{30}$，不招占$p_2 = \frac{25}{30}$

$$\mathrm{Ent}(D^1) = -(\frac{5}{30}\log_2\frac{5}{30} + \frac{25}{30}\log_2\frac{25}{30}) = 0.752$$

举例

1. 计算各个子集的信息熵；2. 加权求和



$\mathrm{Ent}(D^1) = 0.752$　　$\mathrm{Ent}(D^2) = 0.918$

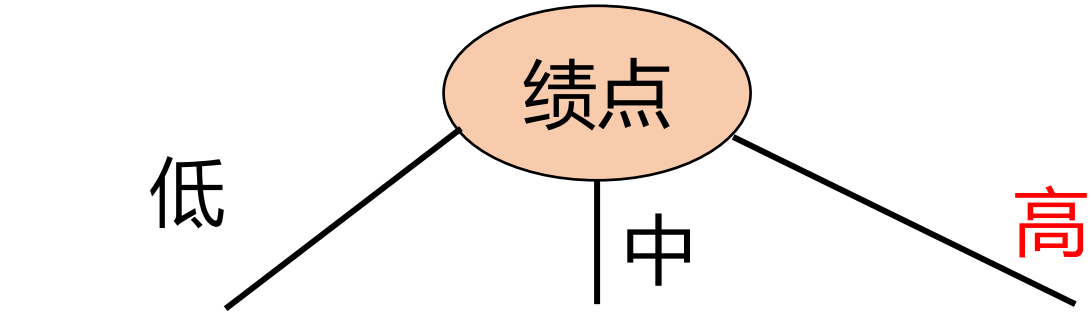
绩点=中的数据集包含30个训练样本，

招的占$p_1 = \frac{20}{30}$，不招占$p_2 = \frac{10}{30}$

$$\mathrm{Ent}(D^2) = -\left(\frac{20}{30}\log_2\frac{20}{30} + \frac{10}{30}\log_2\frac{10}{30}\right) = 0.918$$

举例

1. 计算各个子集的信息熵；2. 加权求和



$\mathrm{Ent}(D^1) = 0.752$　$\mathrm{Ent}(D^2) = 0.918$　$\mathrm{Ent}(D^3) = 0.544$

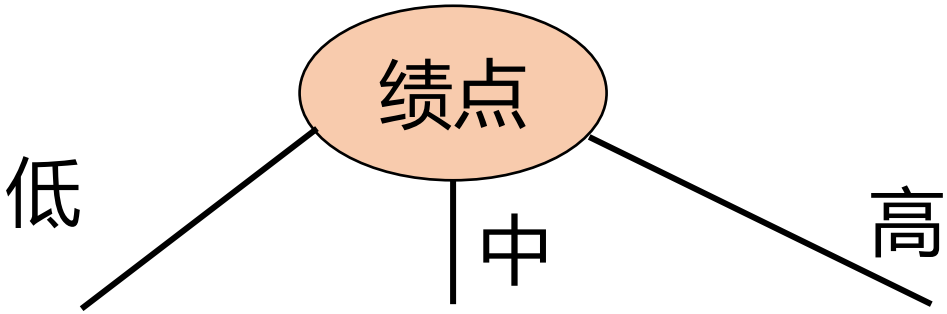
绩点=高的数据集包含30个训练样本，

招的占$p_1 = \frac{35}{40}$，不招占$p_2 = \frac{5}{40}$

$$\mathrm{Ent}(D^3) = -\left(\frac{35}{40}\log_2\frac{35}{40} + \frac{5}{40}\log_2\frac{5}{40}\right) = 0.544$$

举例

1. 计算各个子集的信息熵；2. 加权求和



$\mathrm{Ent}(D^1) = 0.752$　$\mathrm{Ent}(D^2) = 0.918$　$\mathrm{Ent}(D^3) = 0.544$

$|D^1| = 30$ 个数据，占比 $\frac{|D^{30}|}{|D|} = \frac{30}{100}$

$|D^2| = 30$ 个数据，占比 $\frac{|D^2|}{|D|} = \frac{30}{100}$

$|D^3| = 40$ 个数据，占比 $\frac{|D^3|}{|D|} = \frac{40}{100}$

$$\sum_{n=1}^{N} \frac{|D^n|}{|D|} \mathrm{Ent}(D^n) = \frac{30}{100} \times 0.752 + \frac{30}{100} \times 0.918 + \frac{40}{100} \times 0.544$$

$$= 0.719$$

信息增益

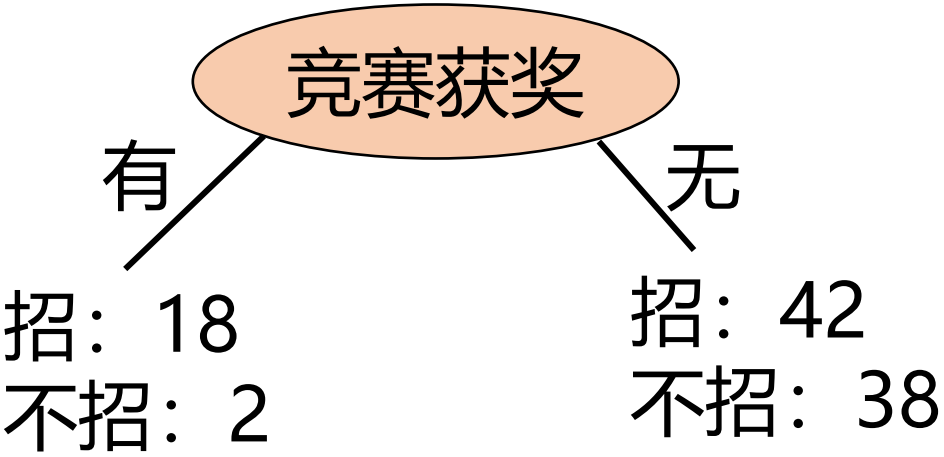
信息增益=当前数据集的信息熵-属性划分后的信息熵

依据绩点对数据进行划分所带来的信息增益：

$$\begin{aligned}\text{Gain}(D,\ \text{绩点}) &= \text{Ent}(D) - \sum_{n=1}^{N} \frac{|D^n|}{|D|}\text{Ent}(D^n) \\ &= 0.971 - 0.719 \\ &= 0.252\end{aligned}$$

举例

1. 计算各个子集的信息熵；2. 加权求和



$$\mathrm{Ent}(D^1) = -\left(\frac{18}{20}\log_2\frac{18}{20} + \frac{2}{20}\log_2\frac{2}{20}\right) = 0.469$$

$$\mathrm{Ent}(D^2) = -\left(\frac{42}{80}\log_2\frac{42}{80} + \frac{38}{80}\log_2\frac{38}{80}\right) = 0.99$$

$$\mathrm{Gain}(D, \text{竞赛获奖}) = \mathrm{Ent}(D) - \sum_{n=1}^{N}\frac{|D^n|}{|D|}\mathrm{Ent}(D^n)$$

$$= 0.971 - 0.896$$

$$= 0.075$$

举例

$$\text{Gain}(D, \text{绩点}) = 0.252$$

$$\text{Gain}(D, \text{竞赛获奖}) = 0.075$$

特征“绩点”的信息增益最大，其被选为划分特征

选择标准

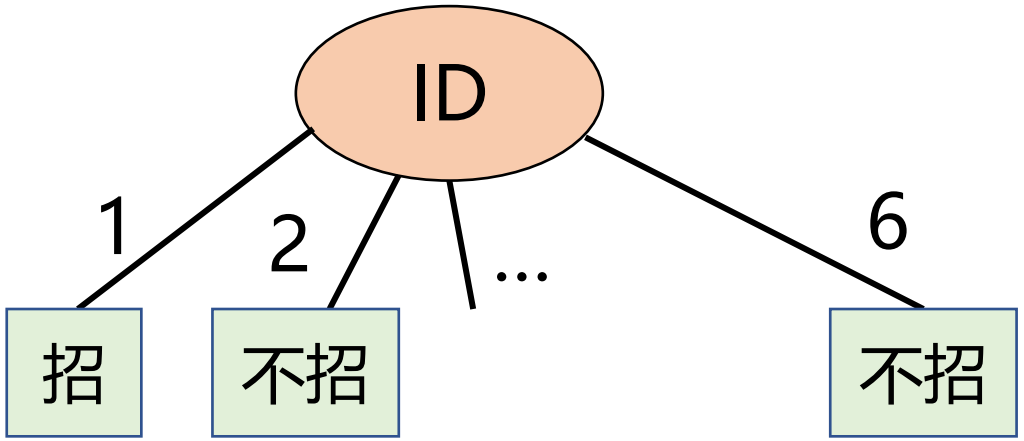
- 信息增益
- 信息增益率
- 基尼系数

信息增益率

信息增益的缺陷：倾向于选择取值比较多的特征，

即特征本身信息熵比较小

ID	招聘与否
1	1
2	0
3	1
4	0
5	1
6	0



熵为0

- 启发：不仅要考虑属性带来的信息增益，
  还要考虑特征本身的信息熵

信息增益率

特征a的固有值 (intrinsic value)：

$$\mathrm{IV}(a) = -\sum_{v=1}^{V} \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|}$$

- 表征特征a的不确定性
- 例如，绩点低的占比0.3，绩点中的占比0.4，绩点高的占比0.3，则

$$\mathrm{IV}(\text{绩点}) = -(0.3 \times \log_2 0.3 + 0.4 \times \log_2 0.4 + 0.3 \times \log_2 0.3)$$

特征a的信息增益率：$\mathrm{Gain\_ratio}(D, a) = \frac{\mathrm{Gain}(D,a)}{\mathrm{IV}(a)}$

选择标准

- 信息增益
- 信息增益率
- 基尼系数

基尼系数

- 假定当前样本集合D中第k类样本所占的比例为p_k
- 基尼系数

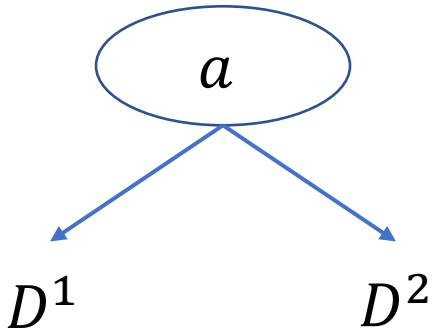
$$\text{Gini}\,(D) = \sum_{k=1}^{K} \sum_{k' \neq k} p_k\, p_{k'} = 1 - \sum_{k=1}^{K} p_k^2$$

- 基尼系数反映了从数据集D中随机抽取两个样本，其标记不一致的概率
- 基尼系数越小，数据纯度越高

属性的基尼系数

CART二叉树（节点的子树数目不超过2）

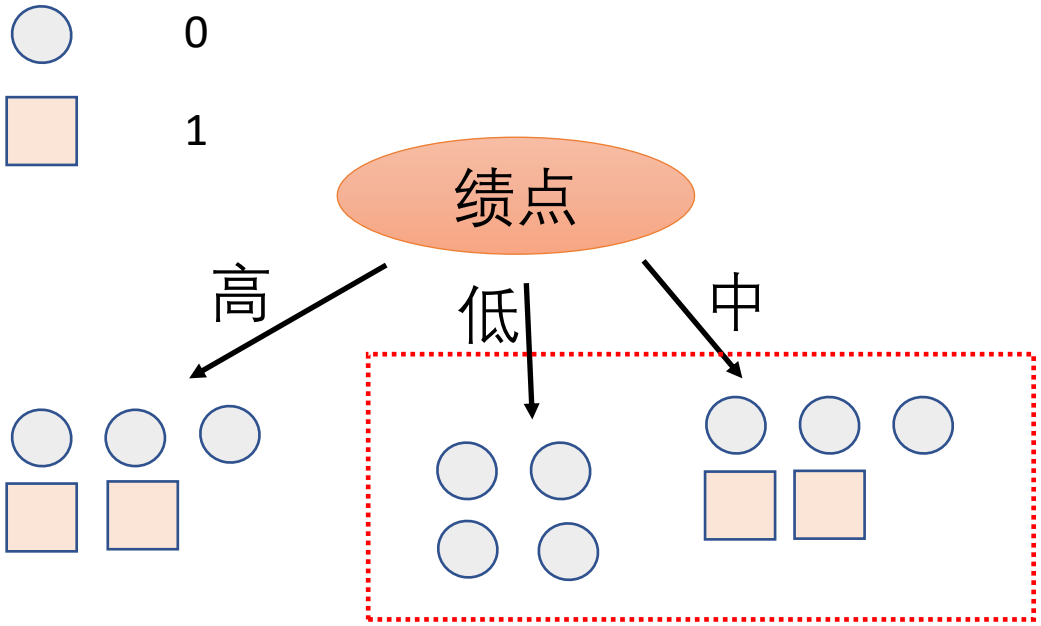
根据特征a是否为某一值将数据集D分成D^1和D^2两个子集



属性a的基尼系数

$$\text{Gini\_index}\,(D,a)=\frac{|D^1|}{|D|}\text{Gini}(D^1)+\frac{|D^2|}{|D|}\text{Gini}(D^2)$$

举例



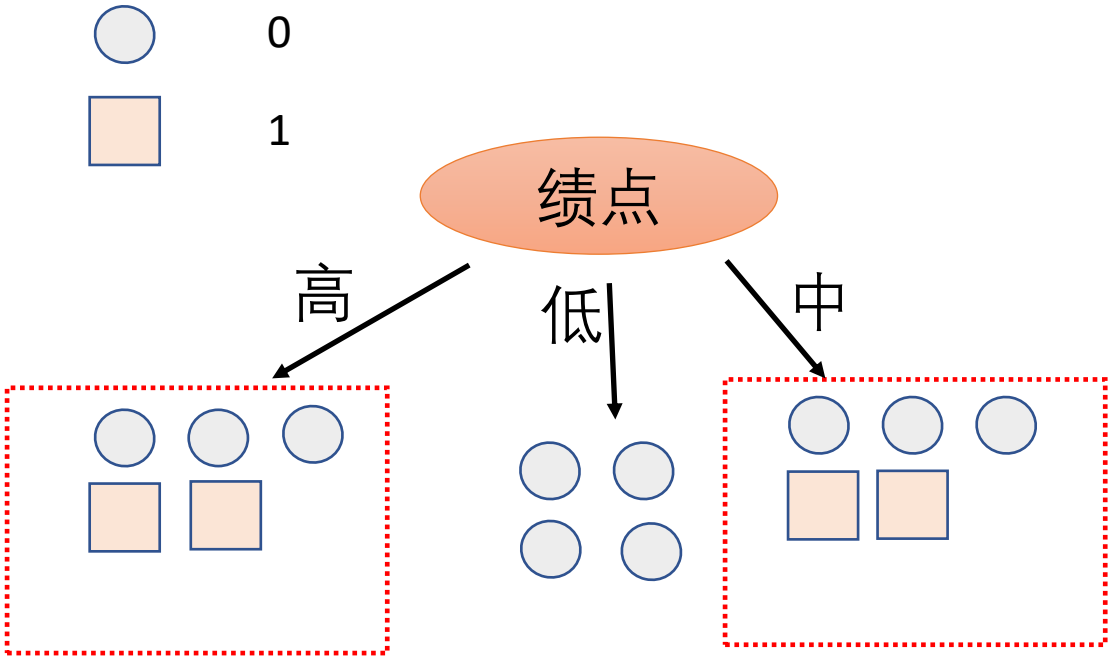
D^1	D^2
高	低，中

$$Gini(D^1) = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 = 0.48 \quad Gini(D^2) = 1 - \left(\frac{7}{9}\right)^2 - \left(\frac{2}{9}\right)^2 = 0.35$$

$$Gini\_index^1(D,a) = \frac{|D^1|}{|D|}Gini(D^1) + \frac{|D^2|}{|D|}Gini(D^2)$$

$$= \frac{5}{14} * 0.48 + \frac{9}{14} * 0.35 = 0.4$$

举例



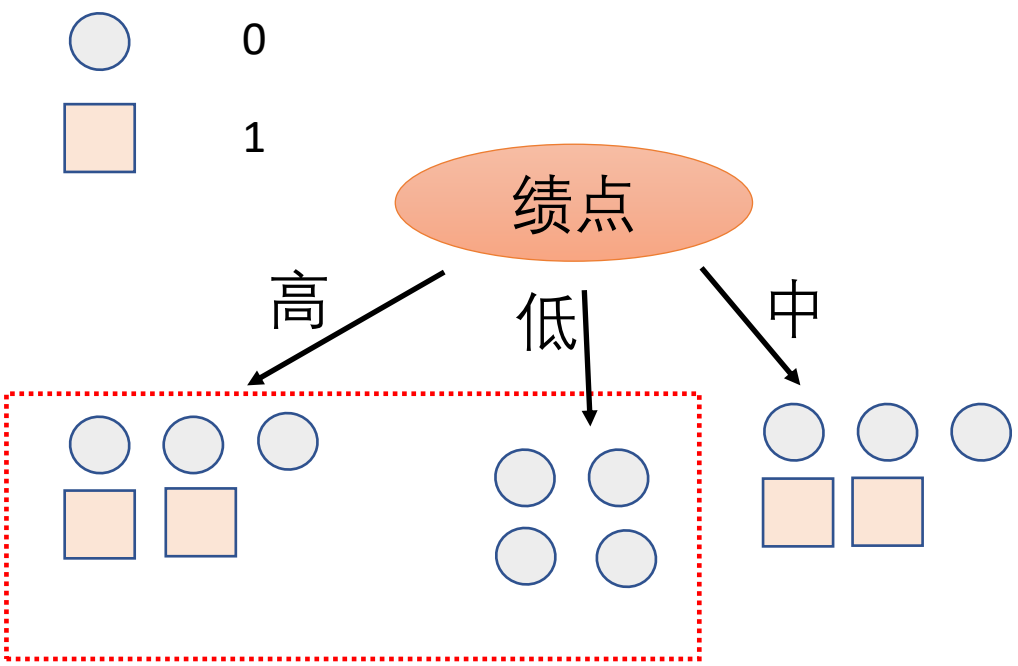
D^1	D^2
高	低，中
低	高，中

$$Gini(D^1) = 1 - \left(\frac{0}{4}\right)^2 - \left(\frac{4}{4}\right)^2 = 0 \qquad Gini(D^2) = 1 - \left(\frac{6}{10}\right)^2 - \left(\frac{4}{10}\right)^2 = 0.51$$

$$Gini\_index^2(D,a) = \frac{|D^1|}{|D|}Gini(D^1) + \frac{|D^2|}{|D|}Gini(D^2)$$

$$= \frac{4}{14} * 0 + \frac{10}{14} * 0.51 = 0.36$$

举例



D^1	D^2
高	低，中
低	高，中
中	高，低

$$Gini(D^1) = 1 - \left(\frac{7}{9}\right)^2 - \left(\frac{2}{9}\right)^2 = 0.3456 \qquad Gini(D^2) = 1 - \left(\frac{6}{10}\right)^2 - \left(\frac{4}{10}\right)^2 = 0.48$$

$$Gini\_index^3(D,a) = \frac{|D^1|}{|D|}Gini(D^1) + \frac{|D^2|}{|D|}Gini(D^2)$$

$$= \frac{5}{14} * 0.3456 + \frac{9}{14} * 0.48 = 0.432$$

$$Gini\_index(D,a) = \min\{Gini\_index^1(D,a), Gini\_index^2(D,a), Gini\_index^3(D,a)\}$$

随堂小测

决策树构建过程中，选择特征的主要依据是（）

A. 特征的准确率

B. 特征的方差大小

C. 信息增益或基尼系数

D. 特征与标签的相关性

内容

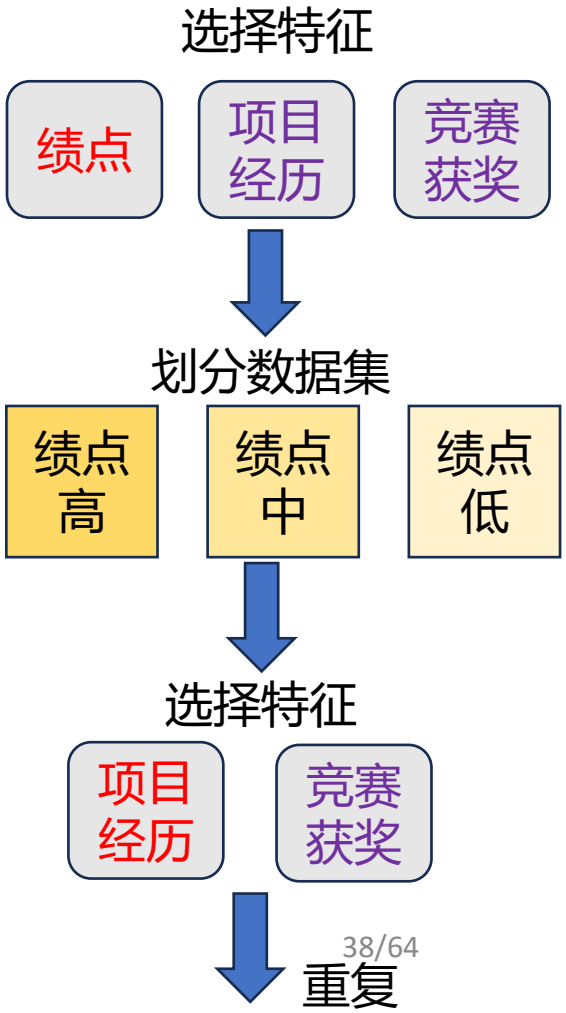
决策树的构建

- 信息增益 (ID3)
- 信息增益率 (C4.5)
- 基尼系数 (CART)

决策树的构建

从根结点开始，通过选择最优特征逐步划分数据集，直到满足停止条件（如达到最大深度、节点纯度足够高或数据集不能再分）

1. 选择最优特征：从当前数据集中选择一个特征，使得数据集的划分效果最好
2. 划分数据集：根据选择的特征，将数据集划分为多个子集
3. 递归构建子树：对每个子集重复上述过程，直到满足停止条件
4. 生成决策树：最终得到树形结构



伪代码

$Decision\ Tree$(特征集合，数据集合)

If 终结条件满足

    return

选择特征：第j个维度

依据第j维度的特征将数据集分成K个集合$\{D_1, \dots, D_K\}$

For k=1,...K, do

    $Decision\ Tree$(特征集合 − 第j个维度，D_k)

决策树构建

Algorithm 1 决策树学习基本算法

输入:

- 训练集 $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$;
- 属性集 $A = \{a_1, \ldots, a_d\}$.

过程: **函数** TreeGenerate(D, A)

```
1: 生成结点 node;
2: if D 中样本全属于同一类别 C then
3:     将 node 标记为 C 类叶结点; return
4: end if
5: if A = ∅ OR D 中样本在 A 上取值相同 then
6:     将 node 标记叶结点，其类别标记为 D 中样本数最多的类; return
7: end if
8: 从 A 中选择最优划分属性 a_*;
9: for a_* 的每一个值 a_*^v do
10:    为 node 生成每一个分枝; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
11:    if D_v 为空 then
12:        将分枝结点标记为叶结点, 其类别标记为 D 中样本最多的类; return
13:    else
14:        以 TreeGenerate(D_v, A − {a_*}) 为分枝结点
15:    end if
16: end for
```

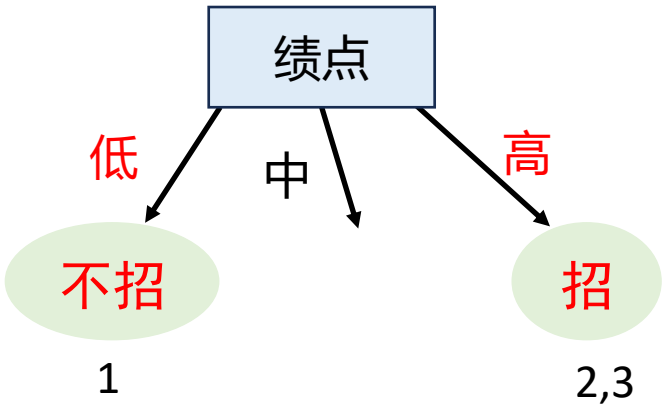
输出: 以 node 为根结点的一棵决策树

决策树构建

2: if 数据集*D* 中样本全属于同一类别 C then
3: 　　将 node 标记为 C 类叶结点; return
4: end if

(1) 检查数据集中所有样本是否属于同一类别

编号	绩点	竞赛获奖	招录与否
1	低	有	不招
2	高	有	招
3	高	无	招
4	中	无	不招
5	中	有	招
6	中	有	不招
7	中	有	招



决策树构建

Algorithm 1 决策树学习基本算法

输入:

- 训练集 $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$;
- 属性集 $A = \{a_1, \ldots, a_d\}$.

过程: **函数** TreeGenerate(D, A)

```
生成结点 node;
if D 中样本全属于同一类别 C then
    将 node 标记为 C 类叶结点; return
end if
if A = ∅ OR D 中样本在 A 上取值相同 then
    将 node 标记叶结点，其类别标记为 D 中样本数最多的类; return
end if
从 A 中选择最优划分属性 a_*;
for a_* 的每一个值 a_*^v do
    为 node 生成每一个分枝; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
    if D_v 为空 then
        将分枝结点标记为叶结点, 其类别标记为 D 中样本最多的类; return
    else
        以 TreeGenerate(D_v, A − {a_*}) 为分枝结点
    end if
end for
```

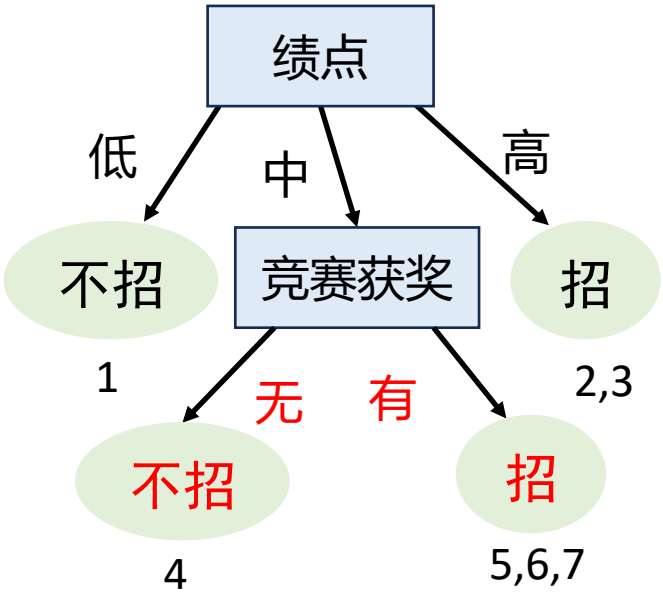
输出: 以 node 为根结点的一棵决策树

决策树构建

5: if A =Ø then
6: 将 node 标记叶结点，其类别标记为数据集D中样本数最多的类; return
7: end if

(2) 检查特征集是否为空

编号	绩点	竞赛获奖	招录与否
1	低	有	不招
2	高	有	招
3	高	无	招
4	中	无	不招
5	中	有	招
6	中	有	不招
7	中	有	招



已使用完两个特征，无法再往下划分

决策树构建

Algorithm 1 决策树学习基本算法

输入:

- 训练集 $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$;
- 属性集 $A = \{a_1, \ldots, a_d\}$.

过程: **函数** TreeGenerate(D, A)

1: **生成结点** node;
2: **if** D 中样本全属于同一类别 C **then**
3: 将 node 标记为 C 类叶结点; **return**
4: **end if**
5: **if** $A = \emptyset$ **OR** D 中样本在 A 上取值相同 **then**
6: 将 node 标记叶结点，其类别标记为 D 中样本数最多的类; **return**
7: **end if**
8: 从 A 中选择最优划分属性 a_*;
9: **for** a_* 的每一个值 a_*^v **do**
10: 为 node 生成每一个分枝; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
11: **if** D_v 为空 **then**
12: 将分枝结点标记为叶结点, 其类别标记为 D 中样本最多的类; **return**
13: **else**
14: 以 TreeGenerate(D_v, $A - \{a_*\}$) 为分枝结点
15: **end if**
16: **end for**

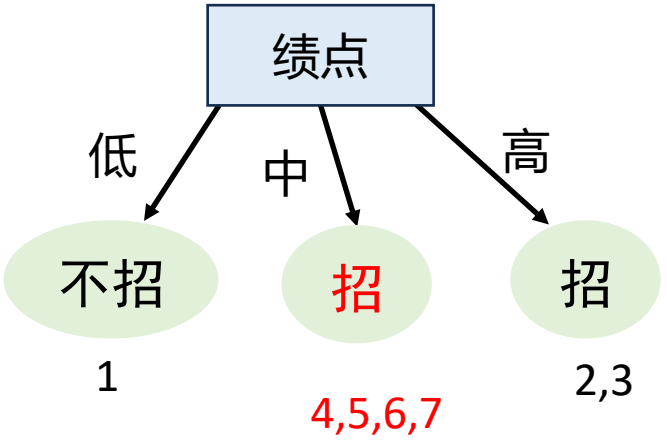
输出: 以 node 为根结点的一棵决策树

决策树构建

5:if 数据集D 中样本在 A 上取值相同 then
6:　将 node 标记叶结点，其类别标记为 D 中样本数最多的类; return
7: end if

（2）检查数据集所有样本的特征值是否相同

编号	绩点	竞赛获奖	招录与否
1	低	有	不招
2	高	有	招
3	高	无	招
4	中	有	招
5	中	有	招
6	中	有	不招
7	中	有	招



ID3决策树

Algorithm 1 决策树学习基本算法

输入:

- 训练集 $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$;
- 属性集 $A = \{a_1, \ldots, a_d\}$.

过程: **函数** TreeGenerate(D, A)

```
1: 生成结点 node;
2: if D 中样本全属于同一类别 C then
3:     将 node 标记为 C 类叶结点; return
4: end if
5: if A = ∅ OR D 中样本在 A 上取值相同 then
6:     将 node 标记叶结点，其类别标记为 D 中样本数最多的类; return
7: end if
8: 从 A 中选择最优划分属性 a_*;
9: for a_* 的每一个值 a_*^v do
10:    为 node 生成每一个分枝; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
11:    if D_v 为空 then
12:        将分枝结点标记为叶结点, 其类别标记为 D 中样本最多的类; return
13:    else
14:        以 TreeGenerate(D_v, A − {a_*}) 为分枝结点
15:    end if
16: end for
```

输出: 以 node 为根结点的一棵决策树

ID3决策树：依据信息增益选择划分属性，即选择信息增益最大的属性作为划分属性

C4.5决策树

Algorithm 1 决策树学习基本算法

输入:

- 训练集 $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$;
- 属性集 $A = \{a_1, \ldots, a_d\}$.

过程: **函数** TreeGenerate(D, A)

1: **生成结点** node;
2: **if** D 中样本全属于同一类别 C **then**
3: 将 node 标记为 C 类叶结点; **return**
4: **end if**
5: **if** $A = \emptyset$ **OR** D 中样本在 A 上取值相同 **then**
6: 将 node 标记叶结点，其类别标记为 D 中样本数最多的类; **return**
7: **end if**
8: 从 A 中选择最优划分属性 a_*;
9: **for** a_* 的每一个值 a_*^v **do**
10: 为 node 生成每一个分枝; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
11: **if** D_v 为空 **then**
12: 将分枝结点标记为叶结点, 其类别标记为 D 中样本最多的类; **return**
13: **else**
14: 以 TreeGenerate(D_v, $A - \{a_*\}$) 为分枝结点
15: **end if**
16: **end for**

输出: 以 node 为根结点的一棵决策树

C4.5决策树：
依据信息增益率
选择划分属性

CART

Algorithm 1 决策树学习基本算法

输入:

- 训练集 $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$;
- 属性集 $A = \{a_1, \ldots, a_d\}$.

过程: **函数** TreeGenerate(D, A)

1: **生成结点** node;
2: **if** D 中样本全属于同一类别 C **then**
3: 将 node 标记为 C 类叶结点; **return**
4: **end if**
5: **if** $A = \emptyset$ **OR** D 中样本在 A 上取值相同 **then**
6: 将 node 标记叶结点，其类别标记为 D 中样本数最多的类; **return**
7: **end if**
8: 从 A 中选择最优划分属性 a_*;
9: **for** a_* 的每一个值 a_*^v **do**
10: 为 node 生成每一个分枝; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
11: **if** D_v 为空 **then**
12: 将分枝结点标记为叶结点, 其类别标记为 D 中样本最多的类; **return**
13: **else**
14: 以 TreeGenerate(D_v, $A - \{a_*\}$) 为分枝结点
15: **end if**
16: **end for**

输出: 以 node 为根结点的一棵决策树

CART决策树：
依据基尼系数
选择划分属性

决策树构建

Algorithm 1 决策树学习基本算法

输入:

- 训练集 $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$;
- 属性集 $A = \{a_1, \ldots, a_d\}$.

过程: **函数** TreeGenerate(D, A)

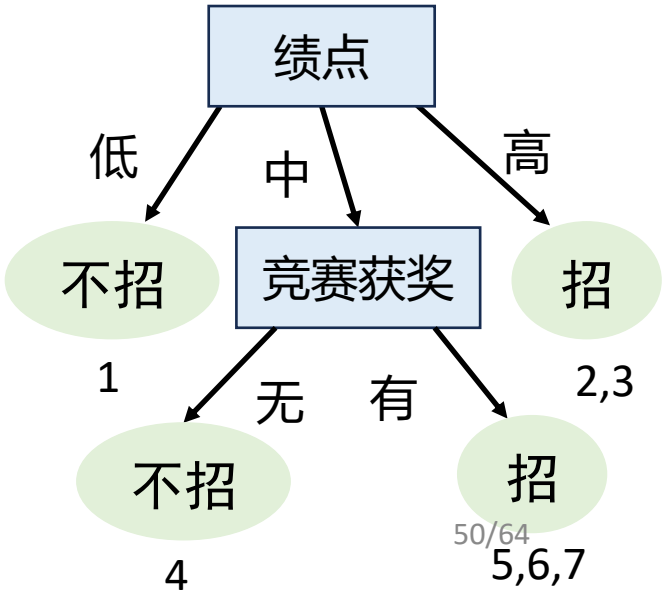
```
1: 生成结点 node;
2: if D 中样本全属于同一类别 C then
3:     将 node 标记为 C 类叶结点; return
4: end if
5: if A = ∅ OR D 中样本在 A 上取值相同 then
6:     将 node 标记叶结点，其类别标记为 D 中样本数最多的类; return
7: end if
8: 从 A 中选择最优划分属性 a_*;
9: for a_* 的每一个值 a_*^v do
10:    为 node 生成每一个分枝; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
11:    if D_v 为空 then
12:        将分枝结点标记为叶结点, 其类别标记为 D 中样本最多的类; return
13:    else
14:        以 TreeGenerate(D_v, A − {a_*}) 为分枝结点
15:    end if
16: end for
```

输出: 以 node 为根结点的一棵决策树

决策树构建

9: for 最优特征a_*的每一个值 a_*^vdo
10: 为 node 生成每一个分枝; 令D_v表示数据集 D 中在取值为 a_*^v的样本子集
11: if D_v为空 then
12: 将分枝结点标记为叶结点,其类别标记为 D中样本最多的类;return
13: else
14: 以 TreeGenerate(D_v,A-$\{a_*\}$)为分枝结点
15: end if
16:end for

编号	绩点	竞赛获奖	招录与否
1	低	有	不招
2	高	有	招
3	高	无	招
4	中	无	不招
5	中	有	招
6	中	有	不招
7	中	有	招



内容

- 基本概念
- 特征选择
- 剪枝

剪枝

- 剪枝：“剪”掉一些叶节点

- 目的：降低模型复杂度，减缓“过拟合”，避免决策分支过多

- 策略：预剪枝和后剪枝

- 预剪枝：在决策树生成过程中
- 后剪枝：在决策树生成之后

预剪枝

提前停止往下划分结点：

1. 树达到预定的高度
2. 节点包含的样本个数小于预定的阈值
3. 信息增益（或信息增益率、基尼系数变化量）小于设定的阈值
4. 划分当前结点不能提升泛化能力

通过预留的验证集来评估：比较划分与不划分的精度

- 划分的精度高，则划分
- 不划分的精度高，则不划分，即将该结点作为叶结点

数据集

训练集

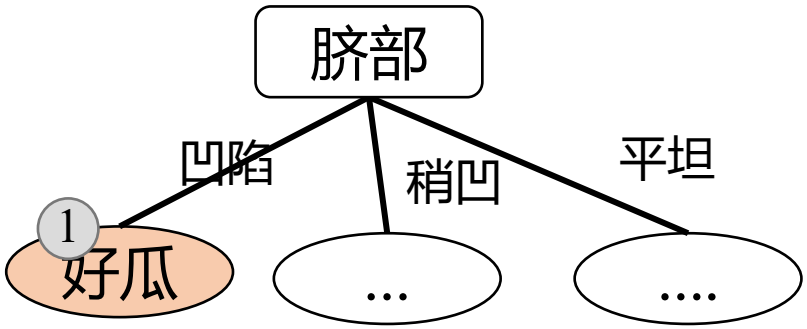
编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

验证集

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否

预剪枝

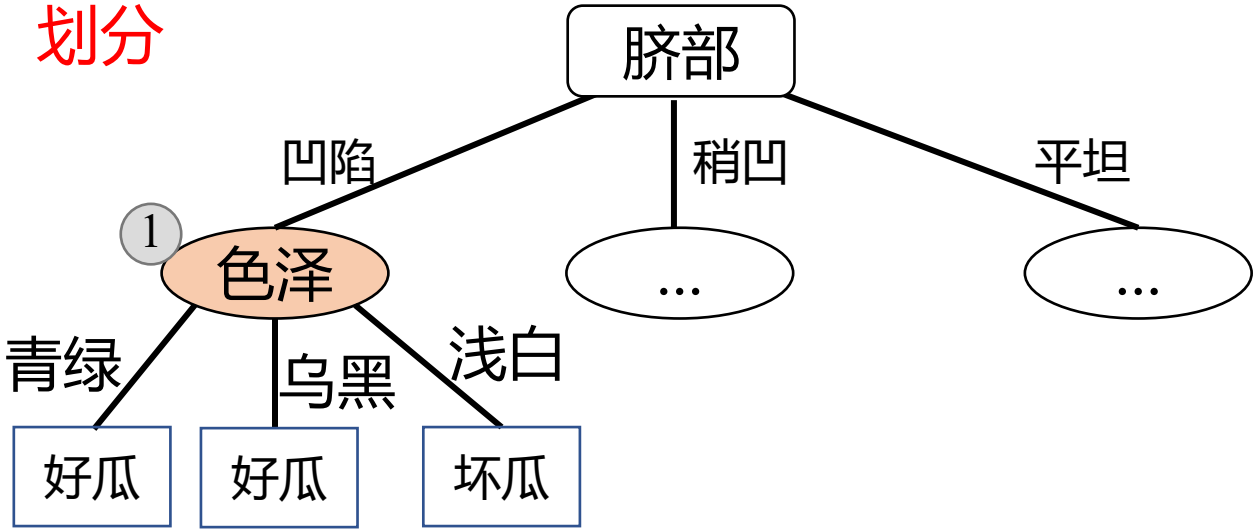
不划分



训练集

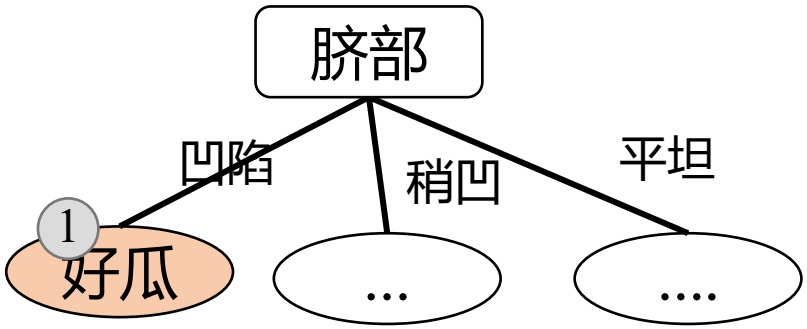
编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

划分



预剪枝

不划分



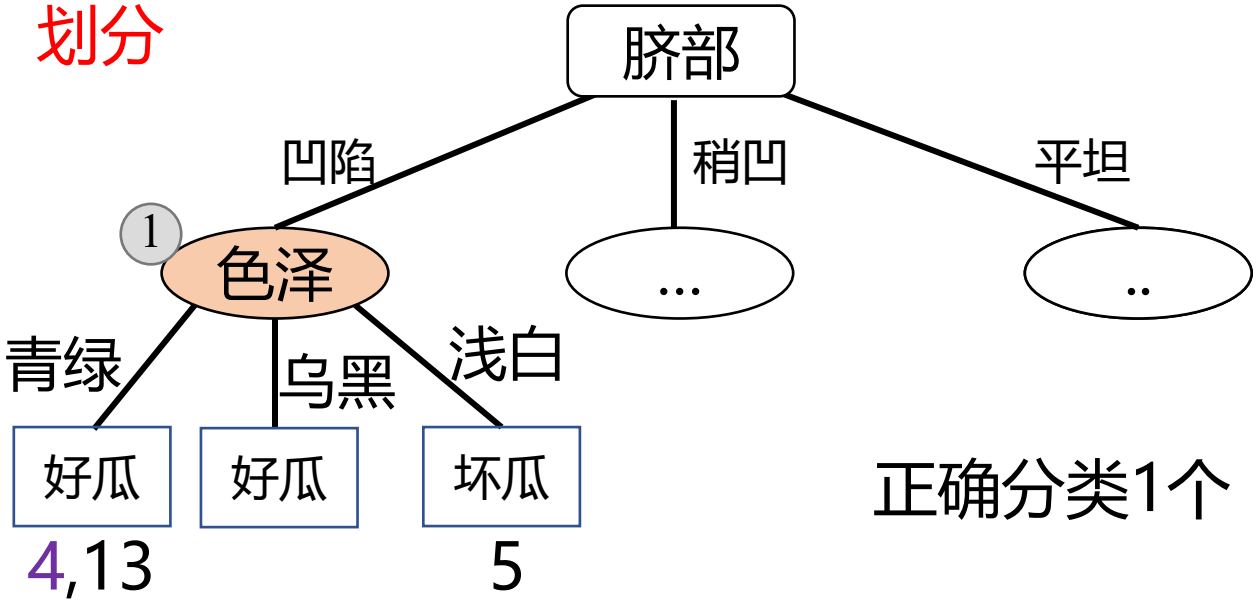
4,5

正确分类2个

验证集

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否

划分



4,13

5

不划分结点1

正确分类1个

预剪枝优缺点

- 优点：
  - 避免了不必要的计算和划分，节省了时间和计算资源
  - 降低过拟合
- 缺点：“贪心”地禁止一些分支展开，可能带来欠拟合

后剪枝

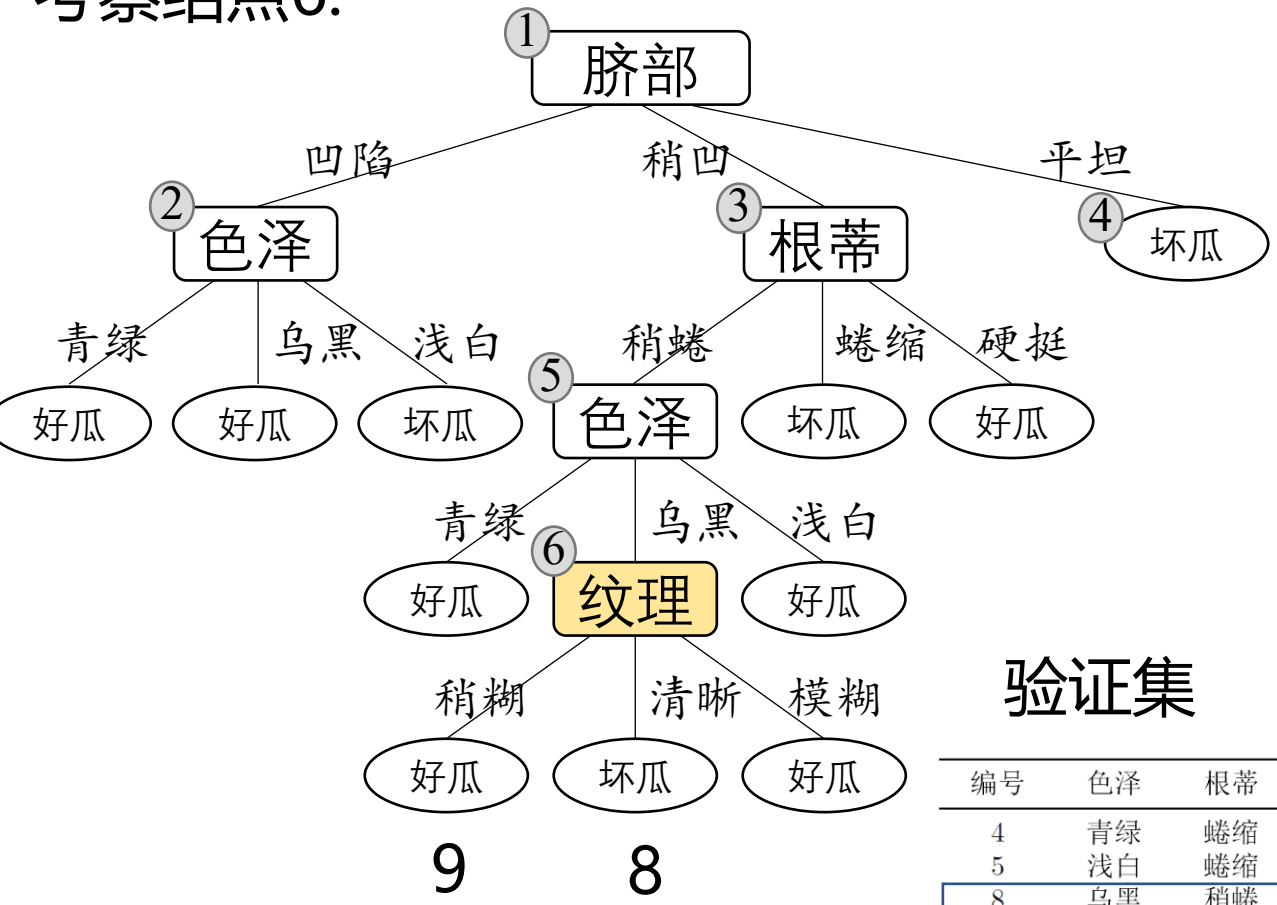
是在决策树构建完成之后，对已经生成的树进行修剪

完整的决策树：
自底向上地考察非叶结点，比较划分和不划分的泛化能力

- 划分的精度高，则划分
- 不划分的精度高，则不划分，即将该结点作为叶结点

后剪枝

考察结点6:



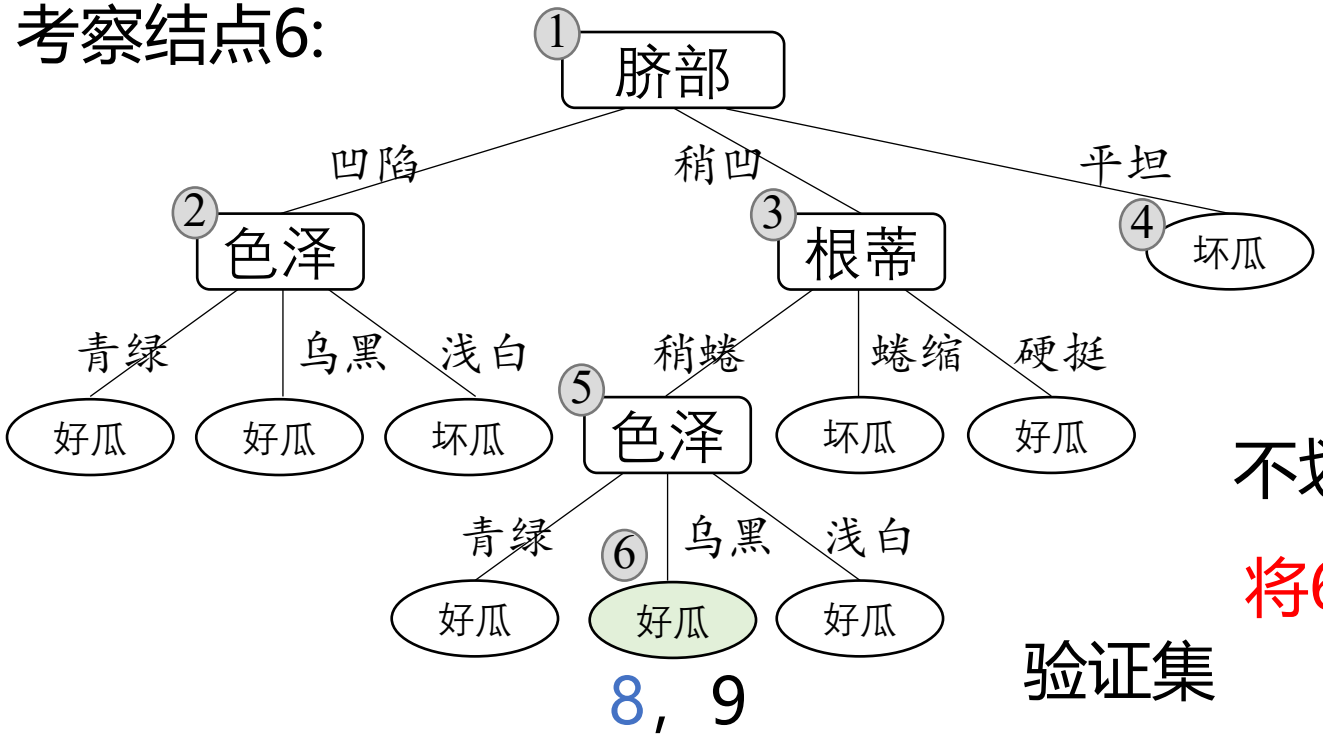
划分结点6：正确0个

验证集

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否

后剪枝

考察结点6:



8，9

不划分结点6：正确1个

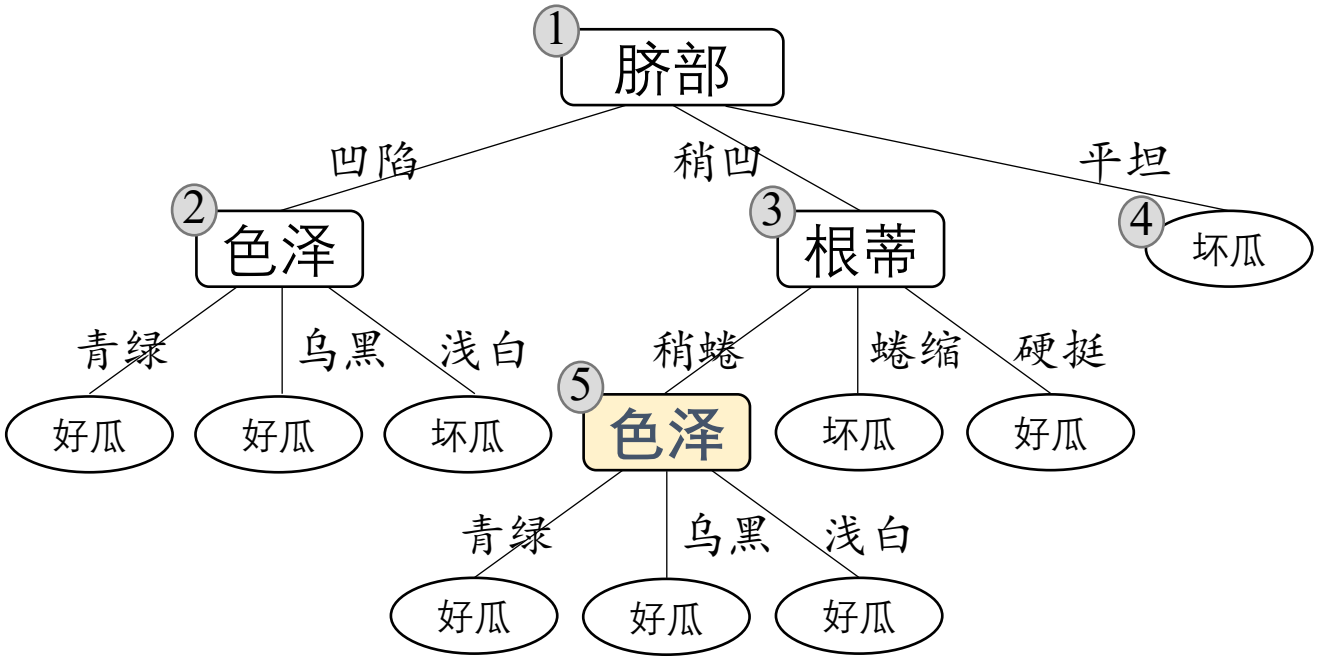
将6结点替换为叶结点

验证集

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否

后剪枝

类似地，考察结点5



后剪枝优缺点

- 优点：
  - 充分地利用训练数据，后剪枝比预剪枝保留了更多的分支，欠拟合风险小，泛化性能往往优于预剪枝决策树
- 缺点：训练时间开销大

随堂小测

决策树算法中，剪枝的主要目的是（）

A. 增加树的深度

B. 提高特征选择效率

C. 减少过拟合

D. 增加模型的解释性

总结

- 决策树基本算法
  - 信息增益（ID3）
  - 增益率（C4.5）
  - 基尼指数（CART）
- 剪枝：减缓过拟合