机器学习

Machine Learning

第四章：决策树

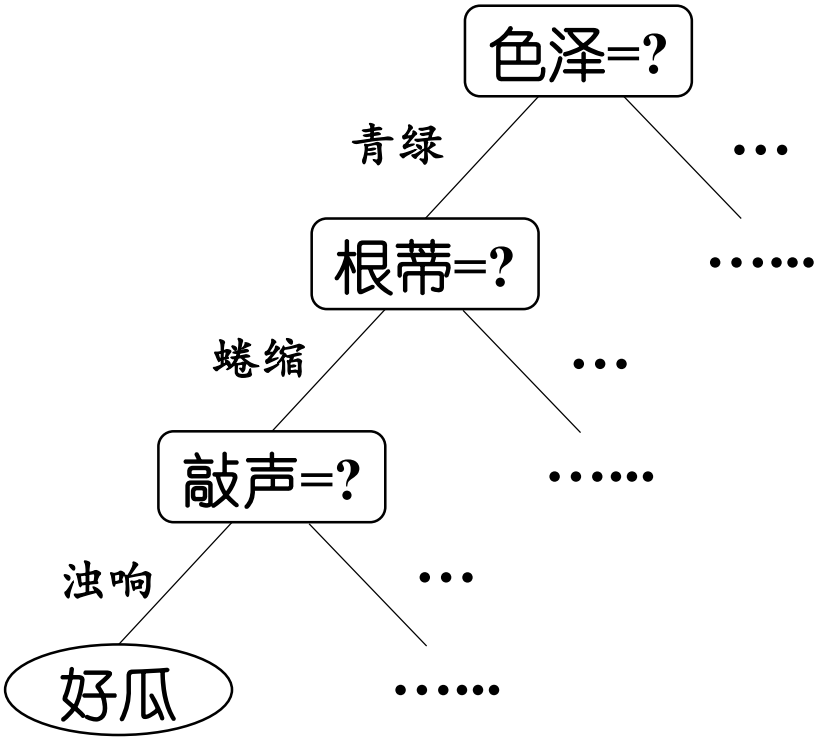
重庆大学计算机学院

大纲

- **基本流程**
- 划分选择
- 剪枝处理
- 连续与缺失值
- 多变量决策树

基本流程

决策树基于树结构来进行预测



主要决策树算法

- ID3 [Quinlan, 1986]

  是最早提出的一种决策树方法，使用上述信息增益的方式建立。

  缺点：只能处理离散型属性，并且对倾向于选择取值较多的属性；

- C4.5 [Quinlan, 1993]

  使用增益率对信息增益进行扩充，以解决偏向取值较多的属性的问题。可以处理连续型属性。

- CART [Breiman et al., 1984]

  CART中用于选择变量的不纯性度量是Gini不存度。构造的是二叉树。

可以用于回归。

基本流程

- **特征选择**：特征选择是指从训练数据中众多的特征中选择一个特征作为当前节点的分裂标准，如何选择特征有着很多不同量化评估标准，从而衍生出不同的决策树算法。
- **决策树生成**： 根据选择的特征评估标准，从上至下递归地生成子节点，直到数据集不可分则停止决策树停止生长。 树结构来说，递归结构是最容易理解的方式。
- **剪枝**：决策树容易过拟合，一般来需要剪枝，缩小树结构规模、缓解过拟合。剪枝技术有预剪枝和后剪枝两种。

决策树学习的目的是为了产生一棵**泛化能力强**，
即**处理未见示例能力强的决策树**

基本流程

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

（1）当前结点包含的样本全部属于同一类别

（2）当前属性集为空，或所有样本在所有属性上取值相同

（3）当前结点包含的样本集合为空

大纲

- 基本流程
- **划分选择**
- 剪枝处理
- 连续与缺失值
- 多变量决策树

划分选择

- 决策树学习的关键在于**如何选择最优划分属性**。一般而言，随着划分过程不断进行，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”(purity)越来越高**
- 经典的属性划分方法：
  - 信息增益 ID3
  - 信息增益率 C4.5
  - 基尼指数 CART

香农熵

- 集合信息的度量方式称为香农熵或者简称为熵(entropy)，来源于信息论之父克劳德·香农。

- 熵定义为信息的期望值。在信息论与概率统计中，熵是表示随机变量不确定性的度量。如果待分类的事物可能划分在多个分类之中，则符号xi的信息定义为 ：

$$l(x_i) = -log_2 p(x_i)$$

  - 其中p(xi)是选择该分类的概率。

香农熵

- 通过上式，可以得到所有类别的信息。为了计算熵，需要计算所有类别所有可能值包含的信息期望值(数学期望)，通过下面的公式得到：

$$\mathrm{H} = -\sum_{\mathrm{i}=1}^{n} \mathrm{p}(x_i)\, log_2 p(x_i)$$

- 其中n是分类的数目。

香农熵的性质

- 信息熵是信息的期望值，描述信息的不确定度。熵越大，表明集合信息的混乱程度越高，换句话说，集合信息混沌，其包含信息价值少。

$$H = -\sum_{i=1}^{n} p(x_i)\, log_2 p(x_i)$$

- Q：抛硬币，熵为多少？有何意义？

$$H=-1/2*\log_2 1/2--1/2*\log_2 1/2=1.0$$

划分选择-信息增益

- “信息熵”是度量样本集合纯度最常用的一种指标，假定当前样本集合D中第k类样本所占的比例为 $p_k \quad (K = 1, 2, ..., |\mathcal{Y}|)$ ，则D的信息熵定义为

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

$\mathrm{Ent}(D)$的值越小，则D的纯度越高

- 计算信息熵时约定：若 $p = 0$ ，则 $p \log_2 p = 0$
- $\mathrm{Ent}(D)$的最小值为0，最大值为$\log_2|\mathcal{Y}|$

划分选择-信息增益

- 离散属性 a 有 V 个可能的取值 $\{a^1, a^2, ..., a^V\}$，用 a 来进行划分，则会产生 V 个分支结点，其中第 v 个分支结点包含了 D 中所有在属性 a 上取值为 a^v 的样本，记为 D^v。则可计算出用属性 a 对样本集 D 进行划分所获得的“信息增益”：

$$\mathrm{Gain}(D, a) = \mathrm{Ent}(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} \mathrm{Ent}(D^v)$$

为分支结点权重，样本数越多的分支结点的影响越大

- 一般而言，**信息增益越大**，则意味着使用属性 a 来进行划分所获得的“**纯度提升**”越大
- ID3决策树学习算法[Quinlan, 1986]以信息增益为准则来选择划分属性

划分选择-信息增益

信息增益实例

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
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
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
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
10	青绿	硬挺	清脆	清晰	平坦	软粘	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
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

该数据集包含17个训练样本，$|\mathcal{Y}| = 2$，其中正例占 $p_1 = \frac{8}{17}$，反例占 $p_2 = \frac{9}{17}$，计算得到根结点的信息熵为

$$\mathrm{Ent}(D) = -\sum_{k=1}^{2} p_k \log_2 p_k = -\left(\frac{8}{17}\log_2\frac{8}{17} + \frac{9}{17}\log_2\frac{9}{17}\right) = 0.998$$

划分选择–信息增益

- 以属性“色泽”为例，其对应的 3 个数据子集分别为 D^1（色泽=青绿），D^2（色泽=乌黑），D^3（色泽=浅白）
- 子集 D^1 包含编号为 $\{1, 4, 6, 10, 13, 17\}$ 的 6 个样例，其中正例占 $p_1 = \frac{3}{6}$，反例占 $p_2 = \frac{3}{6}$，D^2、D^3 同理，3 个结点的信息熵为：

$$\begin{aligned}
\mathrm{Ent}(D^1) &= -\left(\tfrac{3}{6}\log_2\tfrac{3}{6} + \tfrac{3}{6}\log_2\tfrac{3}{6}\right) = 1.000 \\
\mathrm{Ent}(D^2) &= -\left(\tfrac{4}{6}\log_2\tfrac{4}{6} + \tfrac{2}{6}\log_2\tfrac{2}{6}\right) = 0.918 \\
\mathrm{Ent}(D^3) &= -\left(\tfrac{1}{5}\log_2\tfrac{1}{5} + \tfrac{4}{5}\log_2\tfrac{4}{5}\right) = 0.722
\end{aligned}$$

- 属性“色泽”的信息增益为

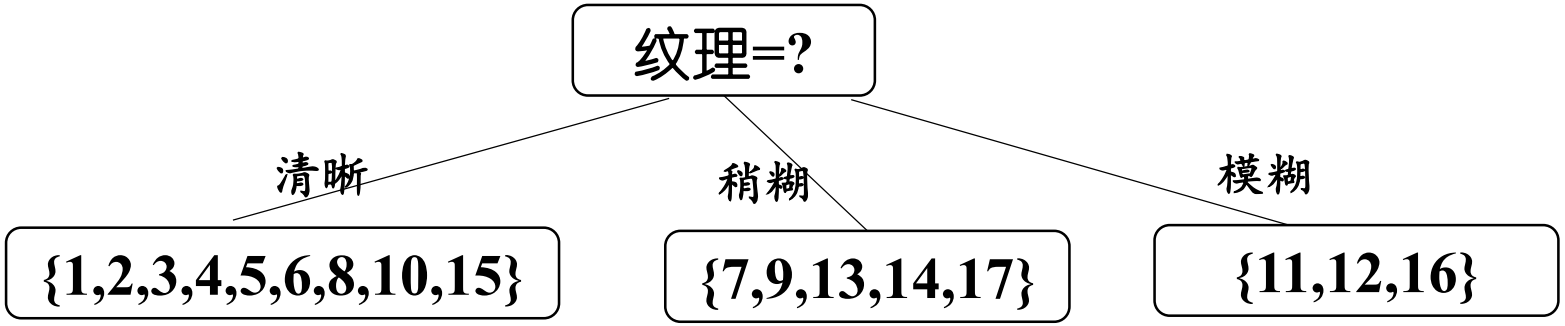
$$\begin{aligned}
\mathrm{Gain}(D, \text{色泽}) &= \mathrm{Ent}(D) - \sum_{v=1}^{3} \frac{|D^v|}{|D|}\mathrm{Ent}(D^v) \\
&= 0.998 - \left(\tfrac{6}{17} \times 1.000 + \tfrac{6}{17} \times 0.918 + \tfrac{5}{17} \times 0.722\right) \\
&= 0.109
\end{aligned}$$

划分选择–信息增益

- 类似的，其他属性的信息增益为

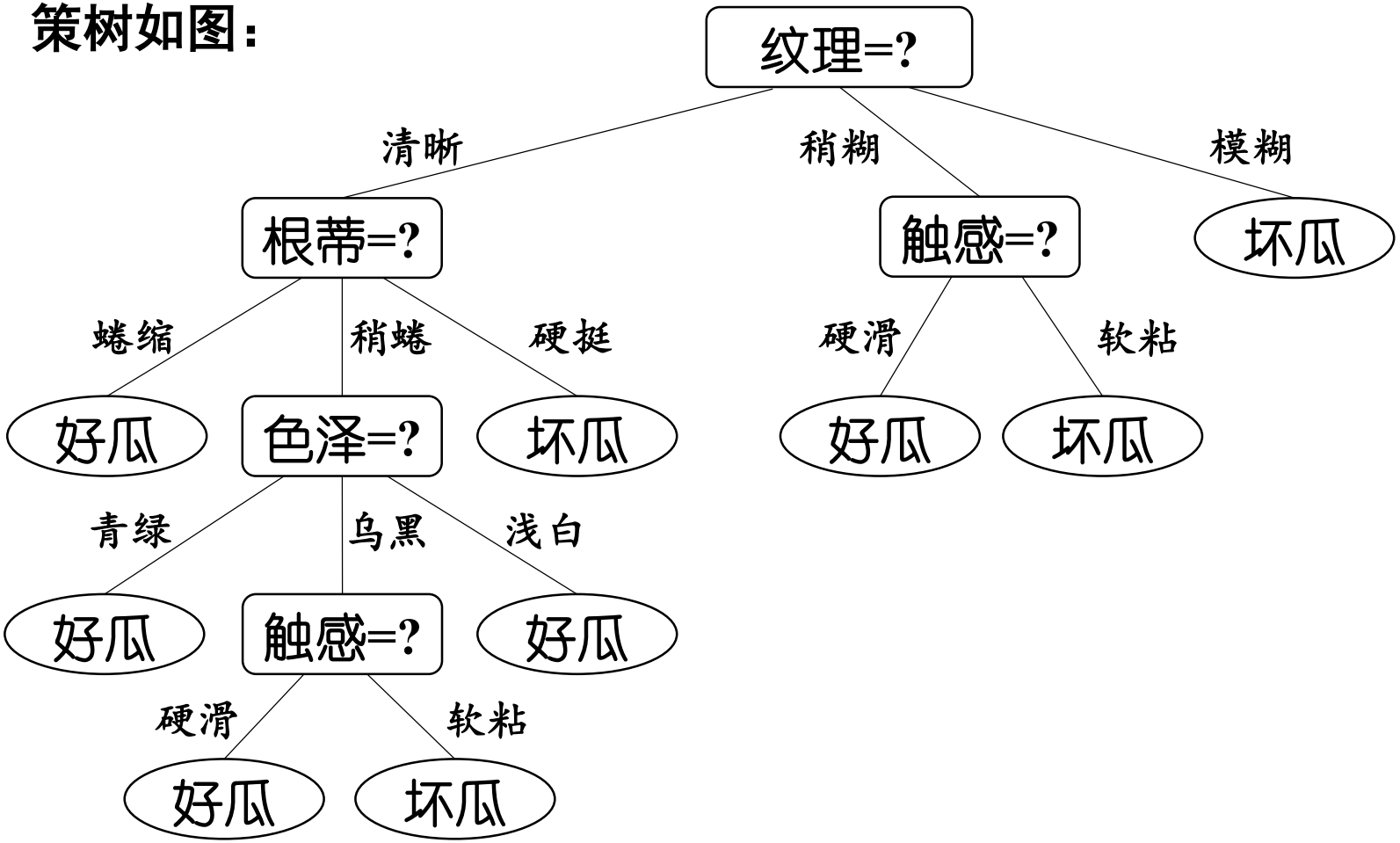
$$\text{Gain}(D, \text{根蒂}) = 0.143 \qquad \text{Gain}(D, \text{敲声}) = 0.141$$
$$\text{Gain}(D, \text{纹理}) = 0.381 \qquad \text{Gain}(D, \text{脐部}) = 0.289$$
$$\text{Gain}(D, \text{触感}) = 0.006$$

- 显然，属性“纹理”的信息增益最大，其被选为划分属性



划分选择-信息增益

- 决策树学习算法将对每个分支结点做进一步划分，最终得到的决策树如图：



划分选择-信息增益

存在的问题

- 若把“编号”也作为一个候选划分属性，则其信息增益一般远大于其他属性。显然，这样的决策树不具有泛化能力，无法对新样本进行有效预测

信息增益对**可取值数目较多**的属性有所偏好

划分选择-增益率

- 增益率定义：

$$\text{Gain\_ratio}(D,a) = \frac{\text{Gain}(D,a)}{\text{IV}(a)}$$

其中

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

称为属性 a 的“固有值”[Quinlan, 1993]，属性 a 的可能取值数目越多（即 V 越大），则 $\text{IV}(a)$ 的值通常就越大

- 属性的固有值（instrisic information）：信息增益率用信息增益 / 固有值，会导致属性的重要性随着固有值的增大而减小（也就是说，如果这个属性本身不确定性就很大，那就越不倾向于选取它），这样算是对单纯用信息增益有所补偿。

属性可取值数目与固有值关系

```
>>> math.log(8,2)
3.0
>>> math.log(16,2)
4.0
>>> iv1 = -0.01*math.log(0.01,2)*100
>>> iv1
6.643856189774724
>>> iv2 = -0.3*math.log(0.3,2)*2-0.4*math.log(0.4,2)
>>> iv2
1.5709505944546684
>>> iv3 = -0.1*math.log(0.1,2)*10
>>> iv3
3.3219280948873626
```

划分选择-增益率

- 存在的问题

  增益率准则对可取值数目较少的属性有所偏好

- C4.5[Quinlan, 1993]使用了一个启发式：先从候选划分属性中找出信息增益高于平均水平的属性，再从中选取增益率最高的
- C5.0：决策树C4.5的商用算法，在内存管理等方面，给出了改进。比如在商用软件SPSS中，就有该算法。

划分选择-基尼指数

- 数据集 D 的纯度可用“基尼值”来度量

$$\mathrm{Gini}(D) = \sum_{k=1}^{|\mathcal{Y}|} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{|\mathcal{Y}|} p_k^2$$

反映了从 D 中随机抽取两个样本，其类别标记不一致的概率

$\mathrm{Gini}(D)$ 越小，数据集 D 的纯度越高

- 属性 a 的基尼指数定义为：

$$\mathrm{Gini\_index}(D, a) = \sum_{v=1}^{V} \frac{|D^v|}{|D|} \mathrm{Gini}(D^v)$$

- 应选择那个使划分后基尼指数最小的属性作为最优划分属性，即

$$a_* = \underset{a \in A}{\mathrm{argmin}}\ \mathrm{Gini\_index}(D, a)$$

- CART [Breiman et al., 1984]采用“基尼指数”来选择划分属性，形成一颗二叉树

数据集

tid	有房者	婚姻状况	年收入	拖欠贷款者
1	是	单身	125K	否
2	否	已婚	100K	否
3	否	单身	70K	否
4	是	已婚	120K	否
5	否	离异	95K	是
6	否	已婚	60K	否
7	是	离异	220K	否
8	否	单身	85K	是
9	否	已婚	75K	否
10	否	单身	90K	是

处理二分类属性

- 以有无房子作为分裂属性计算它的Gini值

有房	无房	拖欠贷款
0	3	是
3	4	否

$$Gini(有房)=1-(3/3)^2-(0/3)^2=0$$

$$Gini(无房)=1-(4/7)^2-(3/7)^2=0.4849$$

$$Gini=0.3\times0+0.7\times0.4898=0.343$$

处理多分类属性

	单身或已婚	离异
否	6	1
是	2	1

Gini(t_1)=1-(6/8)²-(2/8)²=0.375
Gini(t_2)=1-(1/2)²-(1/2)²=0.5
Gini=8/10×0.375+2/10×0.5=0.4

	单身或离异	已婚
否	3	4
是	3	0

Gini(t_1)=1-(3/6)²-(3/6)²=0.5
Gini(t_2)=1-(4/4)²-(0/4)²=0
Gini=6/10×0.5+4/10×0=0.3

	离异或已婚	单身
否	5	2
是	1	2

Gini(t_1)=1-(5/6)²-(1/6)²=0.2778
Gini(t_2)=1-(2/4)²-(2/4)²=0.5
Gini=6/10×0.2778+4/10×0.5=0.3667

处理连续属性

对于连续值处理引进“分裂点”的思想，假设样本集中某个属性共n个连续值，则有n-1个分裂点，每个“分裂点”为相邻两个连续值的均值 (a[i] + a[i+1]) / 2。

	60		70		75		85		90		95		100		120		125		220	
	65		72		80		87		92		97		110		122		172			
	≤	>	≤	>	≤	>	≤	>	≤	>	≤	>	≤	>	≤	>	≤	>		
是	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0		
否	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1		
Gini	0.400		0.375		0.343		0.417		0.400		0.300		0.343		0.375		0.400			

公式小结

信息增益 $\mathrm{Gain}(D,a)=\mathrm{Ent}(D)-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\mathrm{Ent}(D^v)$

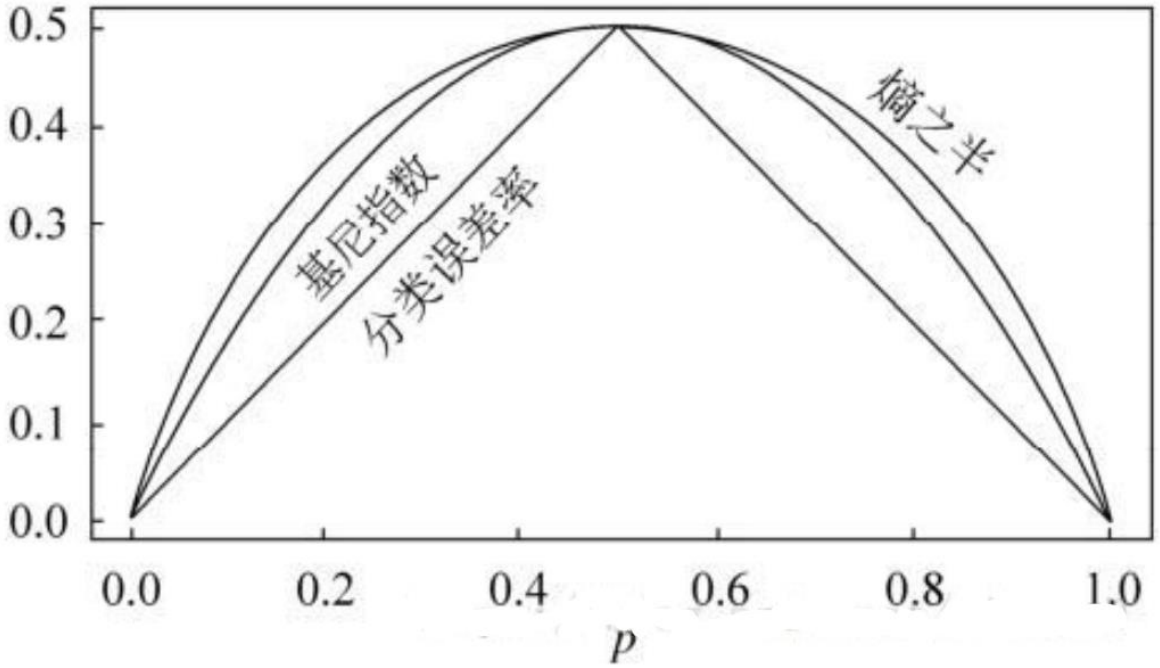
信息增益率

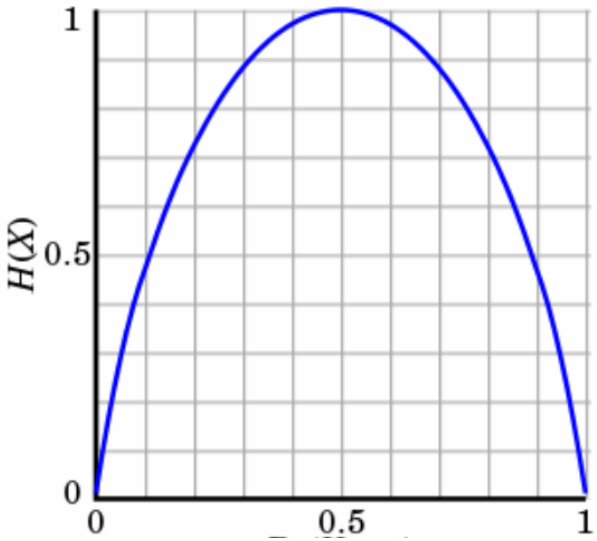
$$\mathrm{Gain\_ratio}(D,a)=\frac{\mathrm{Gain}(D,a)}{\mathrm{IV}(a)} \qquad \mathrm{IV}(a)=-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\log_2\frac{|D^v|}{|D|}$$

Gini不纯度：计算量小

$$\mathrm{Gini\_index}(D,a)=\sum_{v=1}^{V}\frac{|D^v|}{|D|}\mathrm{Gini}(D^v)$$

$$\mathrm{Gini}(D)=\sum_{k=1}^{|\mathcal{Y}|}\sum_{k'\neq k}p_k p_{k'}=1-\sum_{k=1}^{|\mathcal{Y}|}p_k^2$$





$$H(p) = -\sum_{k=1}^{K} p_k log_2(p_k) \qquad \text{Gini}(p) = \sum_{k=1}^{K} p_k(1-p_k) = 1 - \sum_{k=1}^{K} p_k^2$$

熵的公式中有一个log对数，而f(x)=-logx在x=1处一阶泰勒展开，忽略掉高次项，可以得到f(x)≈1-x。这样$p_k \log p_k \approx pk(1-p_k)$，可以看到基尼指数与熵很近似了。

sklearn决策树

sklearn.tree.DecisionTreeClassifier¶

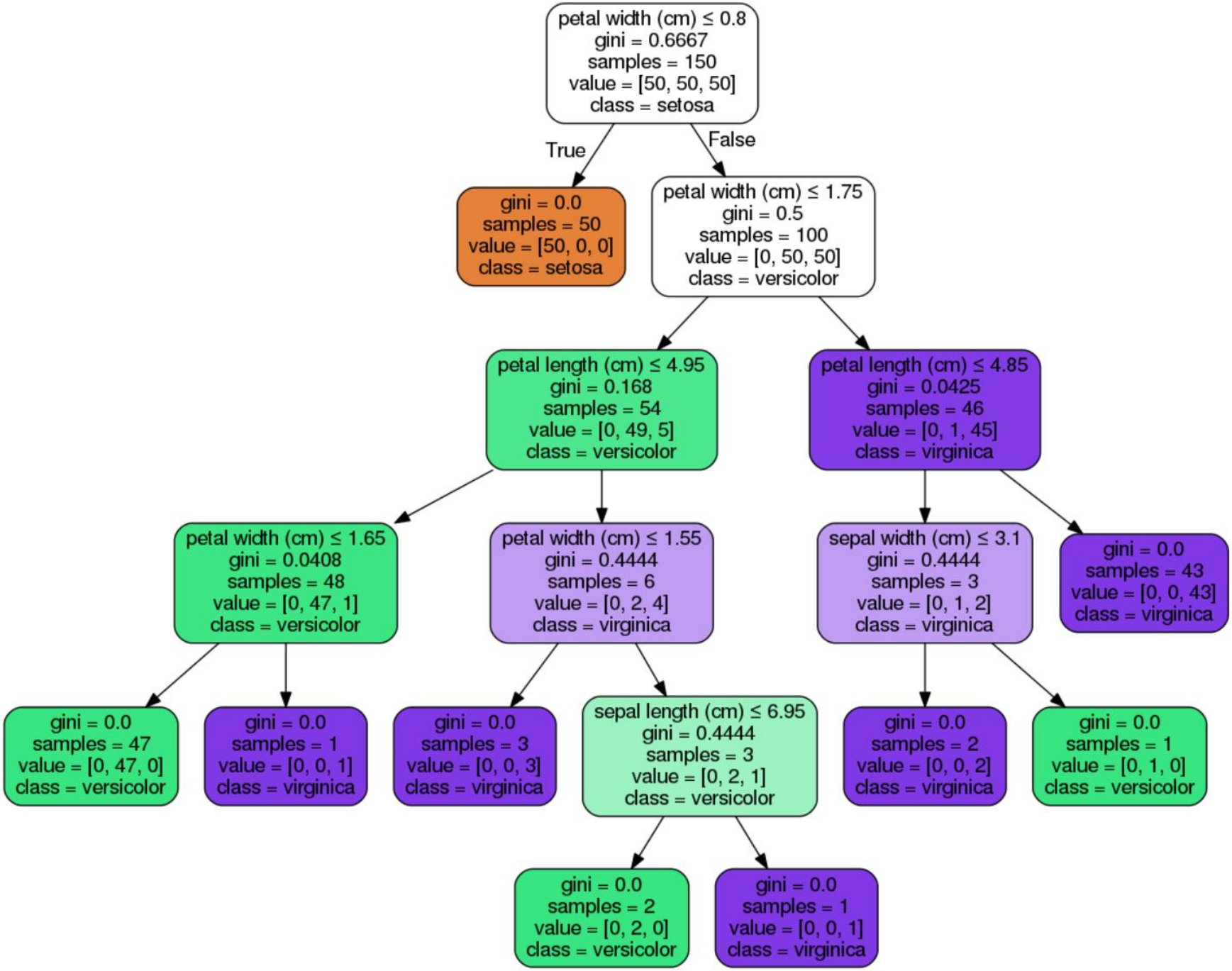
class sklearn.tree. DecisionTreeClassifier (*criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, class_weight=None, presort=False*) [source]

```
>>> from sklearn.datasets import load_iris
>>> from sklearn.model_selection import cross_val_score
>>> from sklearn.tree import DecisionTreeClassifier
>>> clf = DecisionTreeClassifier(random_state=0)
>>> iris = load_iris()
>>> cross_val_score(clf, iris.data, iris.target, cv=10)
...
...
array([ 1.     ,  0.93...,  0.86...,  0.93...,  0.93...,
        0.93...,  0.93...,  1.     ,  0.93...,  1.      ])
```

sklearn决策树

```
from sklearn.externals.six import StringIO
import pydot


dot_data = StringIO()
tree.export_graphviz(clf, out_file=dot_data)
graph = pydot.graph_from_dot_data(dot_data.getvalue())
graph.write_png('iris_simple.png')
```


petal width (cm) ≤ 0.8
gini = 0.6667
samples = 150
value = [50, 50, 50]
class = setosa
True
False
gini = 0.0
samples = 50
value = [50, 0, 0]
class = setosa
petal width (cm) ≤ 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
class = versicolor
petal length (cm) ≤ 4.95
gini = 0.168
samples = 54
value = [0, 49, 5]
class = versicolor
petal length (cm) ≤ 4.85
gini = 0.0425
samples = 46
value = [0, 1, 45]
class = virginica
petal width (cm) ≤ 1.65
gini = 0.0408
samples = 48
value = [0, 47, 1]
class = versicolor
petal width (cm) ≤ 1.55
gini = 0.4444
samples = 6
value = [0, 2, 4]
class = virginica
sepal width (cm) ≤ 3.1
gini = 0.4444
samples = 3
value = [0, 1, 2]
class = virginica
gini = 0.0
samples = 43
value = [0, 0, 43]
class = virginica
gini = 0.0
samples = 47
value = [0, 47, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica
gini = 0.0
samples = 3
value = [0, 0, 3]
class = virginica
sepal length (cm) ≤ 6.95
gini = 0.4444
samples = 3
value = [0, 2, 1]
class = versicolor
gini = 0.0
samples = 2
value = [0, 0, 2]
class = virginica
gini = 0.0
samples = 1
value = [0, 1, 0]
class = versicolor
gini = 0.0
samples = 2
value = [0, 2, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica

大纲

- 基本流程
- 划分选择
- **剪枝处理**
- 连续与缺失值
- 多变量决策树

剪枝处理

- 为什么剪枝
  - “剪枝”是决策树学习算法对付“过拟合”的主要手段
  - 可通过“剪枝”来一定程度避免因决策分支过多，以致于把训练集自身的一些特点当做所有数据都具有的一般性质而导致的过拟合

- 剪枝的基本策略
  - 预剪枝
  - 后剪枝

- 判断决策树泛化性能是否提升的方法
  - 留出法：预留一部分数据用作“验证集”以进行性能评估

剪枝处理

数据集

训练集

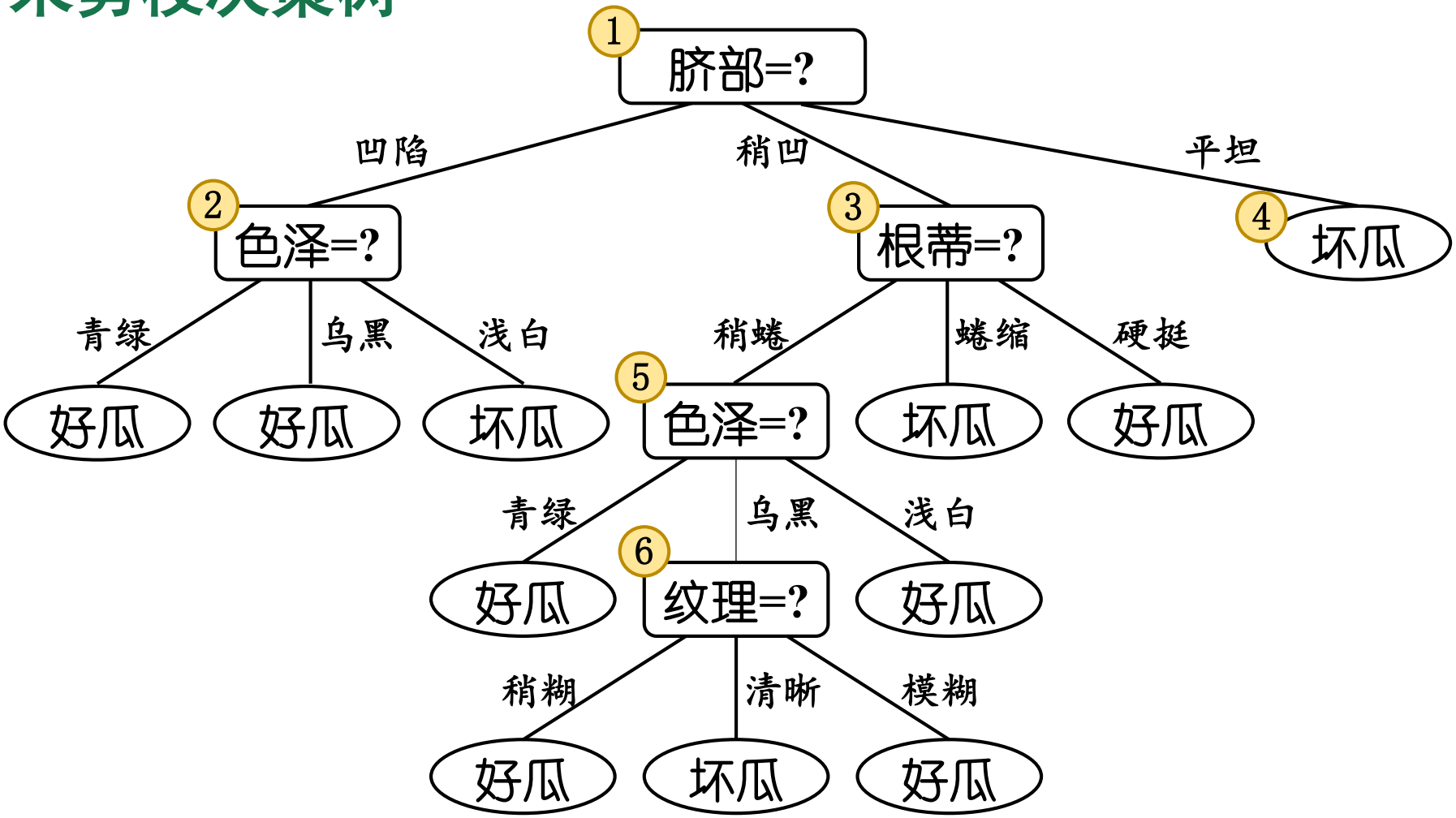
编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

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

剪枝处理

未剪枝决策树


1
脐部=?
凹陷
稍凹
平坦
2
色泽=?
3
根蒂=?
4
坏瓜
青绿
乌黑
浅白
稍蜷
蜷缩
硬挺
好瓜
好瓜
坏瓜
5
色泽=?
坏瓜
好瓜
青绿
乌黑
浅白
6
好瓜
纹理=?
好瓜
稍糊
清晰
模糊
好瓜
坏瓜
好瓜

剪枝处理–预剪枝

- 决策树生成过程中，对每个结点在划分前先进行估计，若当前结点的划分不能带来决策树泛化性能提升，则停止划分并将当前结点记为叶结点，其类别标记为训练样例数最多的类别

- 针对上述数据集，基于信息增益准则，选取属性“脐部”划分训练集。分别计算划分前（即直接将该结点作为叶结点）及划分后的验证集精度，判断是否需要划分。若划分后能提高验证集精度，则划分，对划分后的属性，执行同样判断；否则，不划分

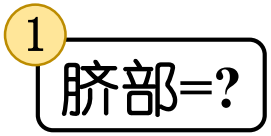
剪枝处理–预剪枝

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

结点1：若不划分，则将其标记为叶结点，类别标记为训练样例中最多的类别，即好瓜。验证集中，$\{4,5,8\}$ 被分类正确，得到验证集精度为 $\frac{3}{7}\times 100\% = 42.9\%$

验证集精度



“脐部=?” 划分前：42.9%

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

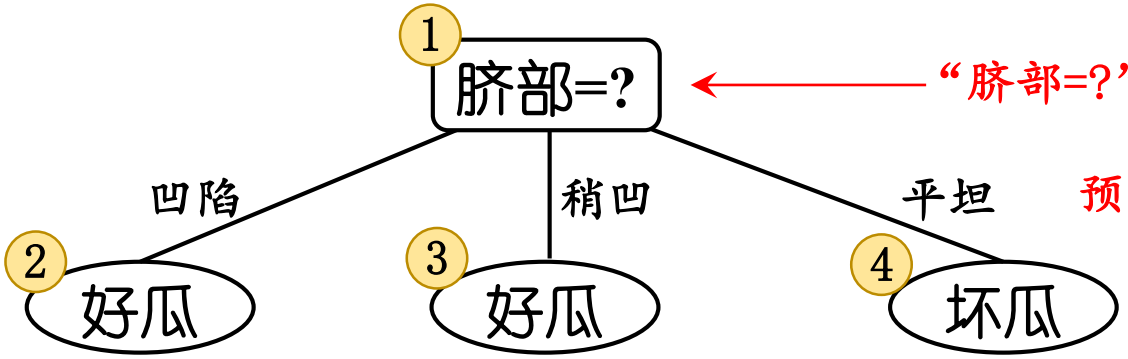
剪枝处理–预剪枝

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

结点1：若划分，根据结点②，③，④的训练样例，将这3个结点分别标记为“好瓜”、“好瓜”、“坏瓜”。此时，验证集中编号为$\{4,5,8,11,12\}$的样例被划分正确，验证集精度为$\frac{5}{7}\times 100\% = 71.4\%$

验证集精度



“脐部=?” 划分前：42.9%

划分后：71.4%

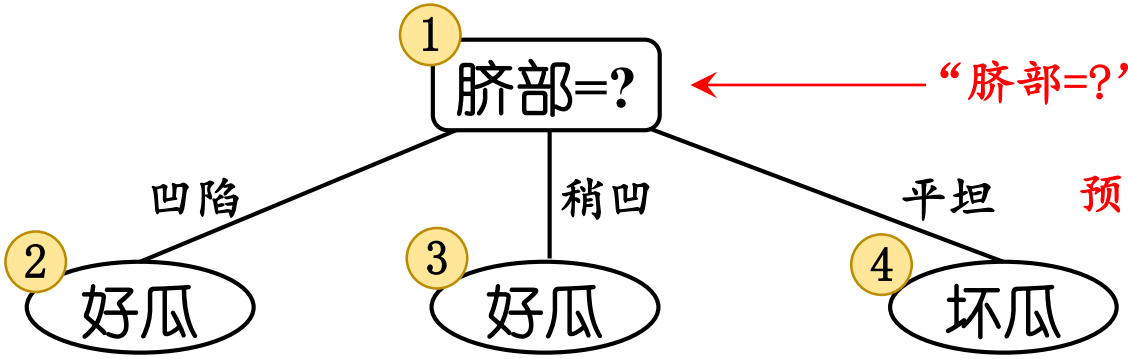
预剪枝决策：划分

剪枝处理–预剪枝

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

对结点②，③，④ 分别进行剪枝判断，结点②，③都禁止划分，结点 ④ 本身为叶子结点。最终得到仅有一层划分的决策树，称为“决策树桩”



验证集精度

“脐部=?” 划分前：42.9%

划分后：71.4%

预剪枝决策：划分

“色泽=?”

验证集精度

划分前：71.4%

划分后：57.1%

预剪枝决策：禁止划分

验证集精度

“根蒂=?” 划分前：71.4%

划分后：71.4%

预剪枝决策：禁止划分

剪枝处理–预剪枝

预剪枝的优缺点

优点

- 降低过拟合风险
- 显著减少训练时间和测试时间开销

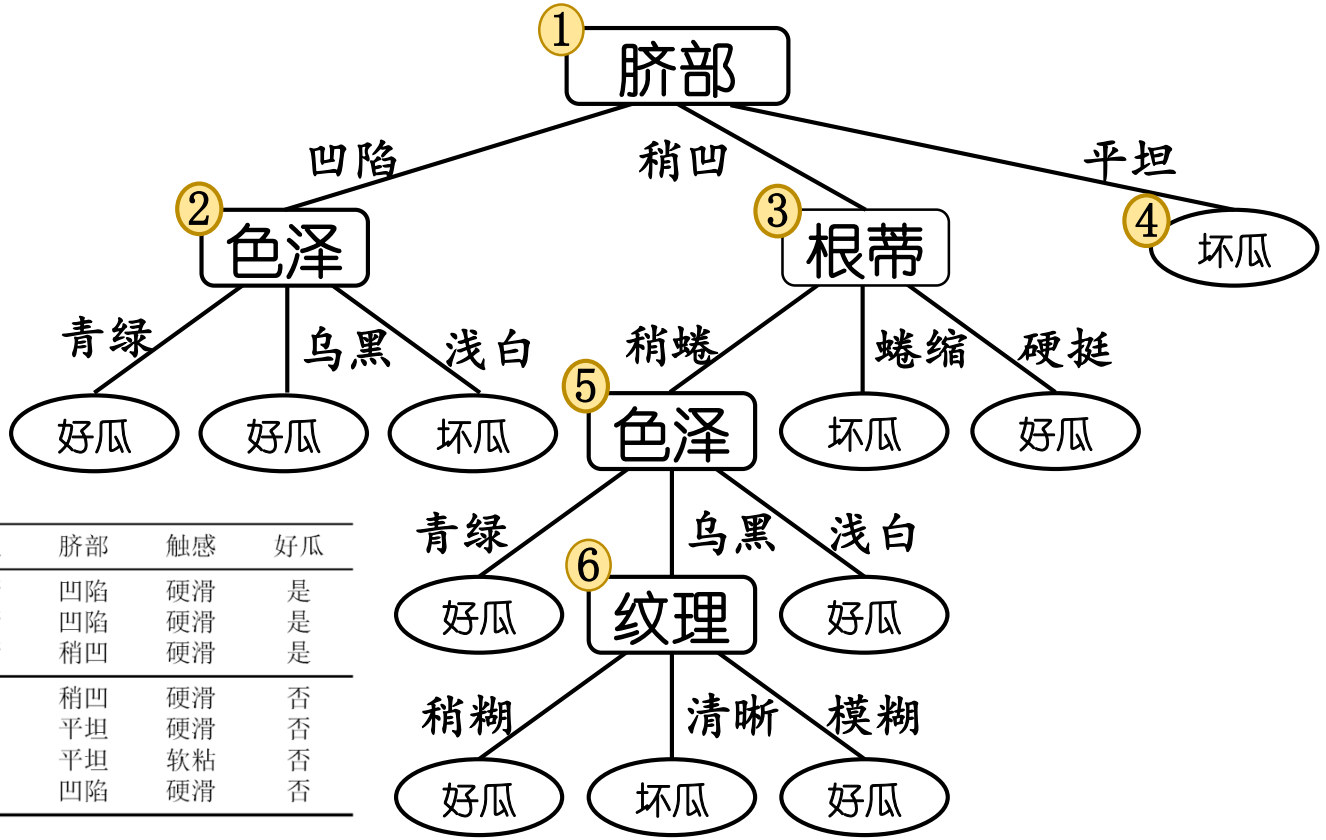
缺点

- 欠拟合风险：有些分支的当前划分虽然不能提升泛化性能，但在其基础上进行的后续划分却有可能导致性能显著提高。预剪枝基于“贪心”本质禁止这些分支展开，带来了欠拟合风险

剪枝处理–后剪枝

- 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，则将该子树替换为叶结点

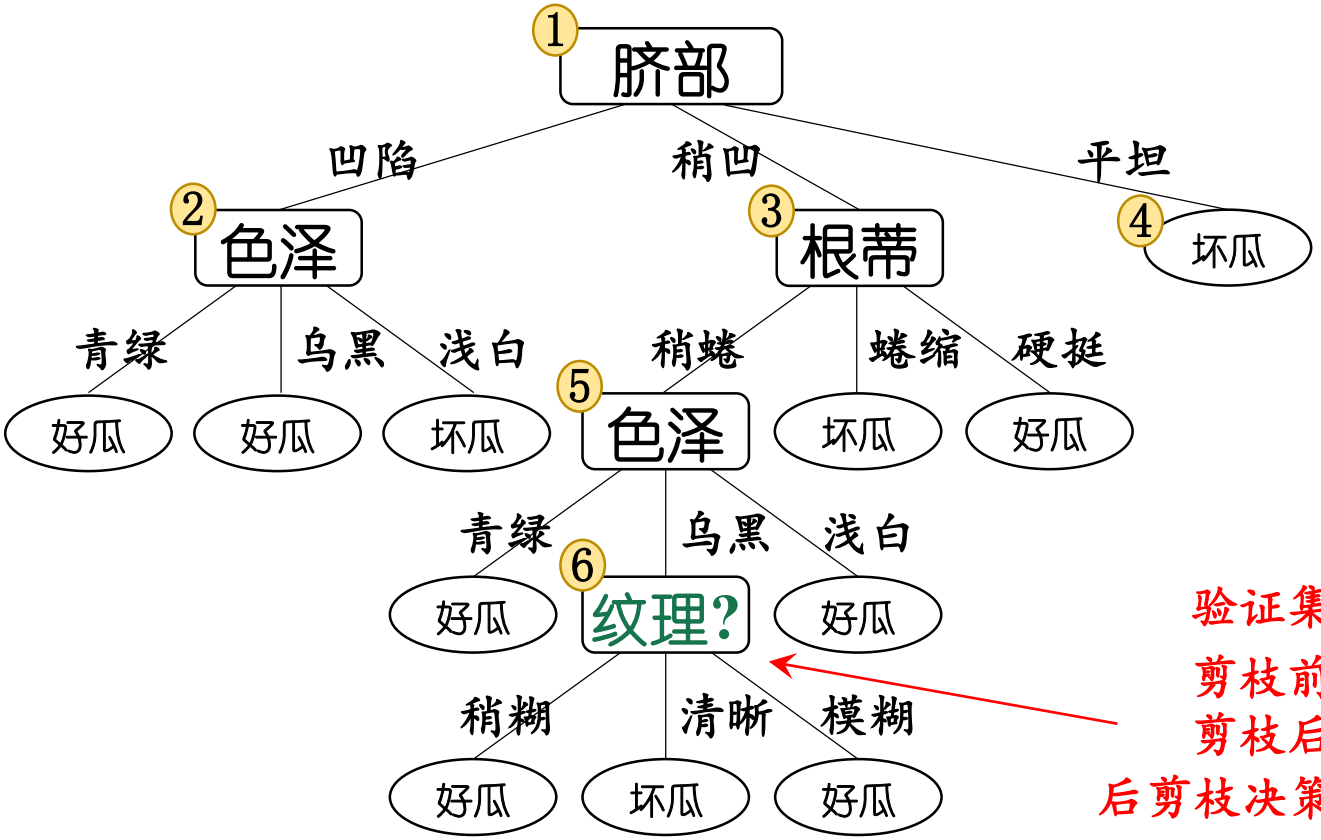
首先生成一棵完整的决策树，该决策树的验证集精度为42.9%



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

剪枝处理-后剪枝

- 首先考虑结点⑥，若将其替换为叶结点，根据落在其上的**训练样本**$\{7,15\}$将其标记为“好瓜”，得到**验证集**精度提高至 57.1% ，则决定剪枝



验证集精度

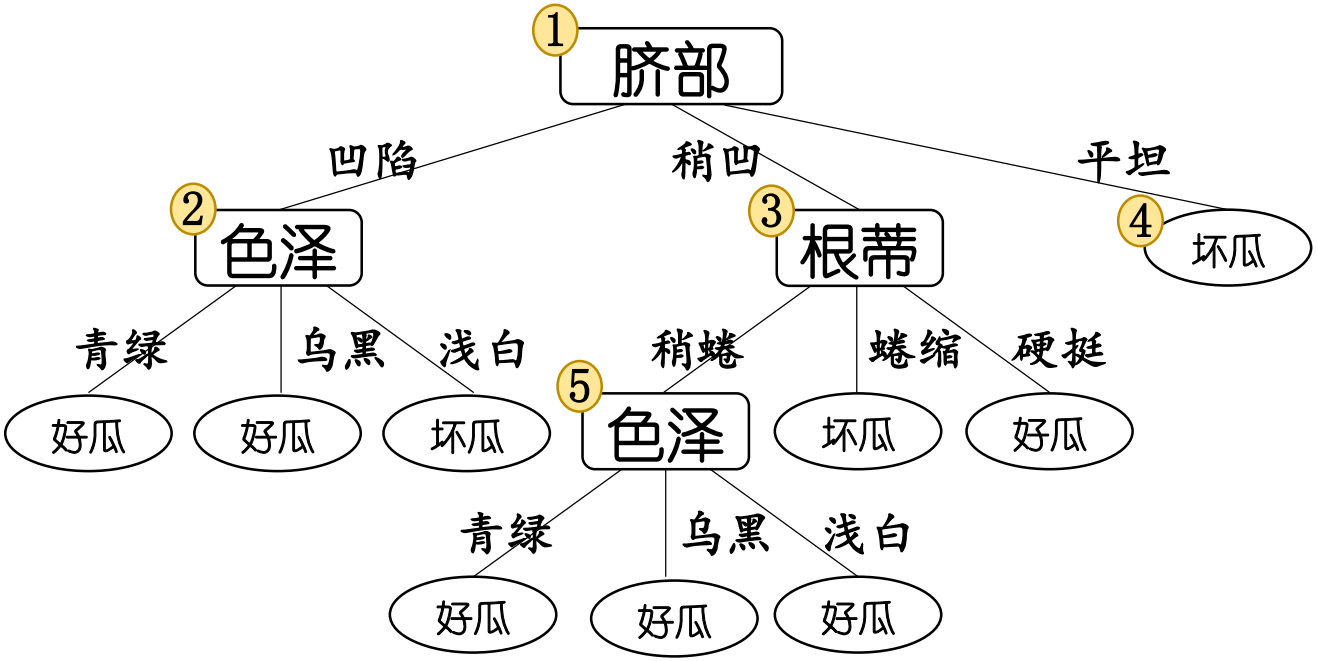
剪枝前：42.9%

剪枝后：57.1%

后剪枝决策：剪枝

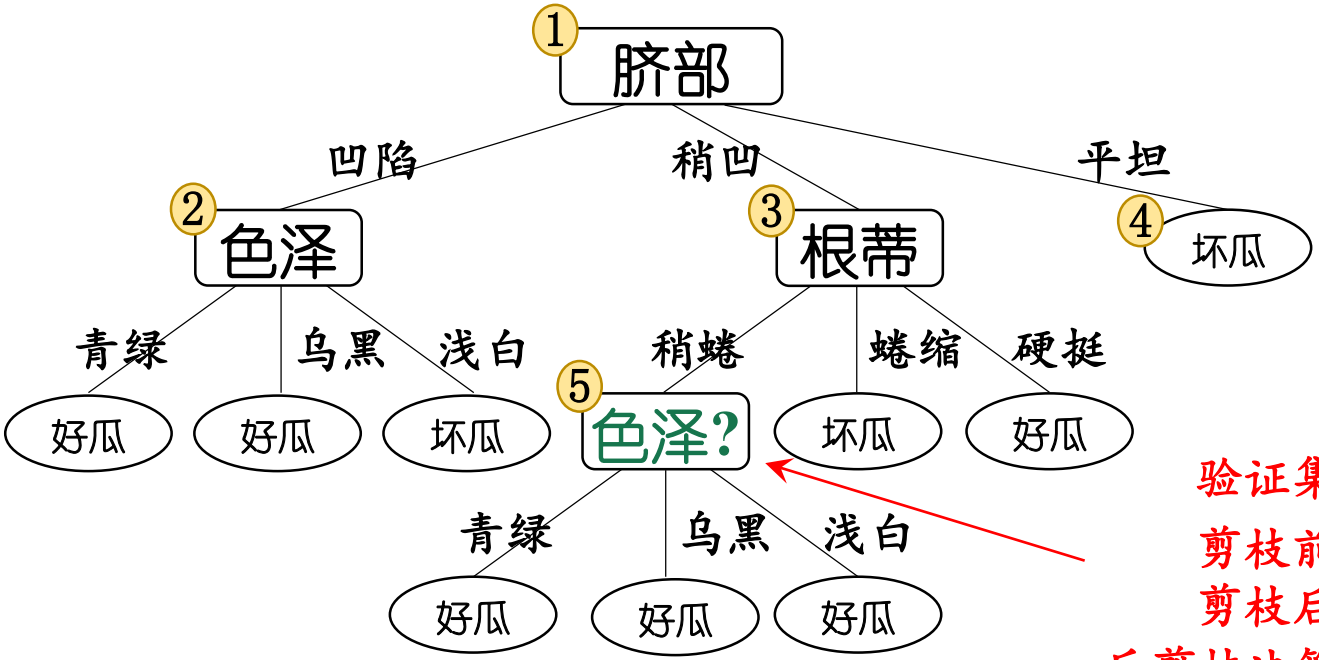
剪枝处理-后剪枝

- 首先考虑结点⑥，若将其替换为叶结点，根据落在其上的训练样本$\{7,15\}$将其标记为“好瓜”，得到验证集精度提高至 57.1% ，则决定剪枝



剪枝处理–后剪枝

- 然后考虑结点⑤，若将其替换为叶结点，根据落在其上的训练样本$\{6,7,15\}$将其标记为“好瓜”，得到验证集精度仍为57.1%，可以不进行剪枝



验证集精度

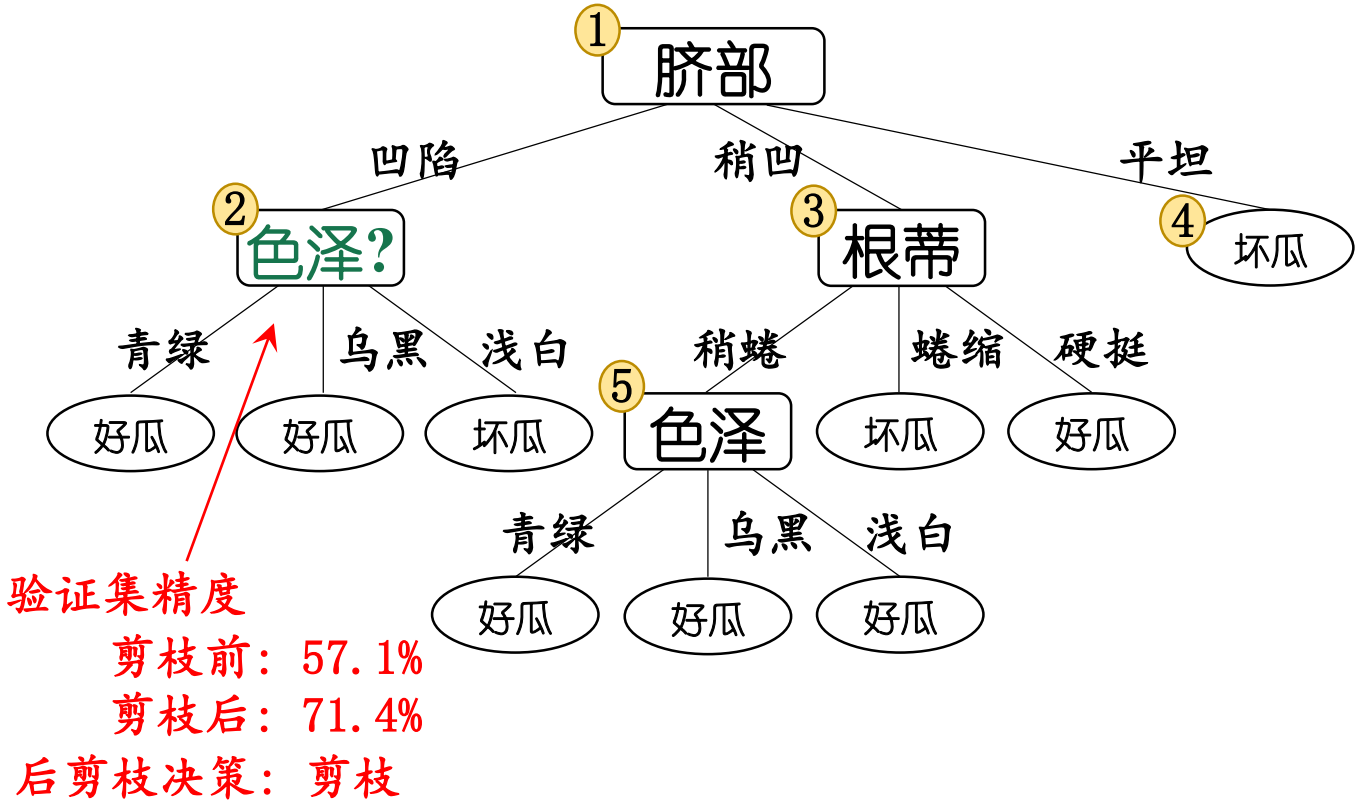
剪枝前：57.1 %

剪枝后：57.1%

后剪枝决策：不剪枝

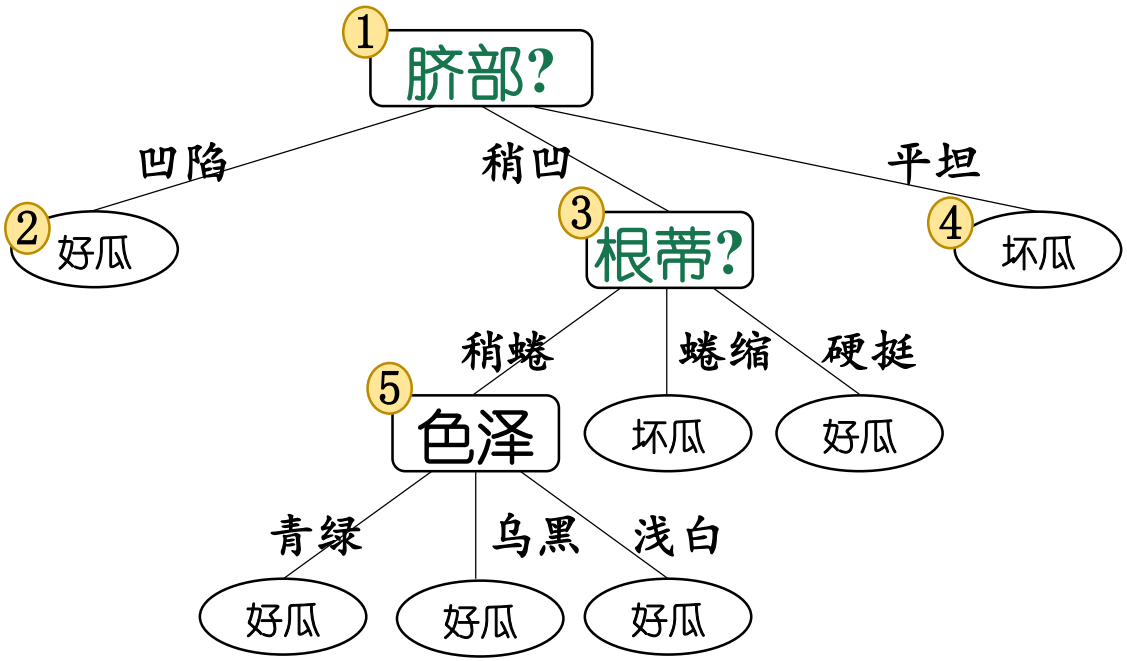
剪枝处理-后剪枝

- 对结点②，若将其替换为叶结点，根据落在其上的训练样本$\{1,2,3,14\}$，将其标记为“好瓜”，得到验证集精度提升至71.4%，则决定剪枝



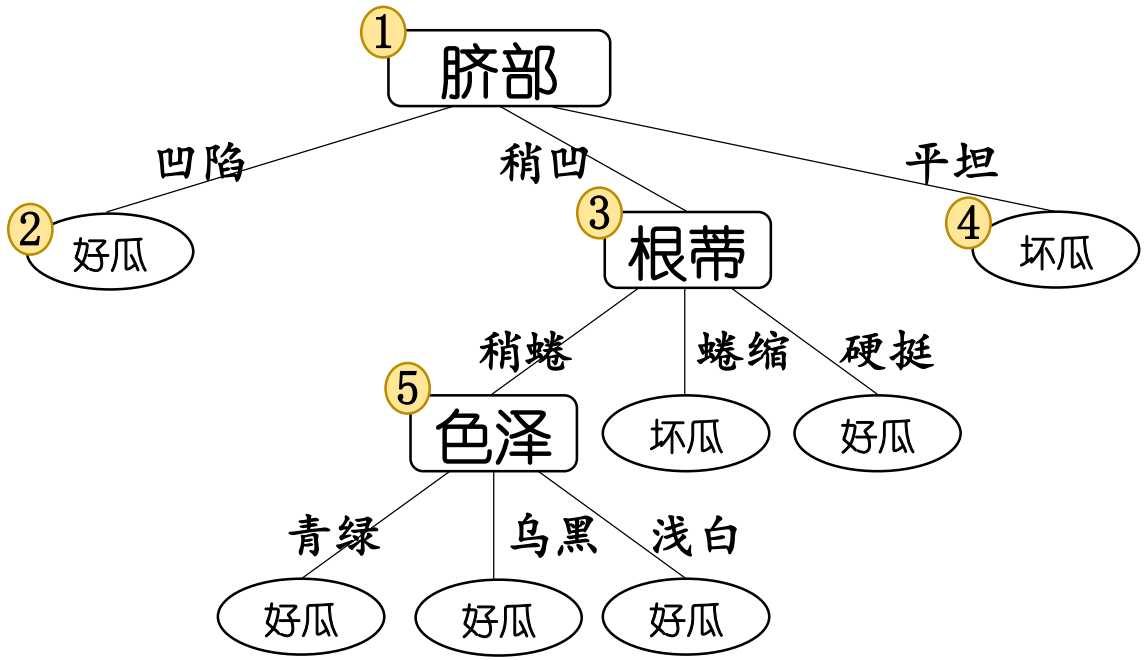
剪枝处理–后剪枝

- 对结点③和① ，先后替换为叶结点，验证集精度均未提升，则分支得到保留



剪枝处理–后剪枝

- 最终基于后剪枝策略得到的决策树如图所示



剪枝处理–后剪枝

后剪枝的优缺点

优点

- 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：后剪枝过程是在生成完全决策树之后进行的，需要自底向上对所有非叶结点逐一考察

大纲

- 基本流程
- 划分选择
- 剪枝处理
- **连续与缺失值**
- 多变量决策树

连续与缺失值 - 连续值处理

- 连续属性离散化(二分法)
  - 第一步：假定连续属性a在样本集D上出现n个不同的取值，从小到大排列，记为 $a^1, a^2, ...a^n$，基于划分点 t，可将 D分为子集 D_t^- 和 D_t^+，其中 D_t^- 包含那些在属性 a上取值不大于t的样本，D_t^+ 包含那些在属性a上取值大于t的样本。考虑包含 $n-1$ 个元素的候选划分点集合

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2} \mid 1 \leq i \leq n-1 \right\}$$

即把区间 $[a^i, a^{i-1})$ 的中位点 $\frac{a^i + a^{i+1}}{2}$作为候选划分点

连续与缺失值 - 连续值处理

- 连续属性离散化(二分法)
  - 第二步：采用离散属性值方法，考察这些划分点，选取最优的划分点进行样本集合的划分

$$\begin{aligned}\mathrm{Gain}(D,a) &= \max_{t\in T_a} \mathrm{Gain}(D,a,t) \\ &= \max_{t\in T_a} \mathrm{Ent}(D) - \sum_{\lambda\in\{-,+\}} \frac{|D_t^\lambda|}{|D|}\mathrm{Ent}(D_t^\lambda)\end{aligned}$$

其中 $\mathrm{Gain}(D,a,t)$ 是样本集 D 基于划分点 t 二分后的信息增益，于是，就可选择使 $\mathrm{Gain}(D,a,t)$ 最大化的划分点

连续与缺失值 - 连续值处理

连续值处理实例

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

对属性“密度”，其候选划分点集合包含16个候选值：

$$T_{\text{密度}} = \{0.244, 0.294, 0.351, 0.381, 0.420, 0.459, 0.518, 0.574, 0.600, 0.621, 0.636, 0.648, 0.661, 0.681, 0.708, 0.746\}$$

可计算其信息增益为0.262，对应划分点为0.381

对属性“含糖量”进行同样处理

与离散属性不同，若当前结点划分属性为连续属性，该属性还可作为其后代结点的划分属性

属性“密度”
信息增益计算

对应划分点：0.381

0.243		是	否
0.245	0.244	4	0
0.343	0.294		
0.36	0.3515		
0.403	0.3815	8	5
0.437	0.42		
0.481	0.459		
0.556	0.5185		
0.593	0.5745		
0.608	0.6005		
0.634	0.621		
0.639	0.6365		
0.657	0.648		
0.666	0.6615		
0.697	0.6815		
0.719	0.708		
0.774	0.7465		

```
>>> import math
>>> EE = -8/17*math.log(8/17,2)-9/17*math.log(9/17,2)
>>> EE
0.9975025463691153
>>> CE = 0 + 13/17*(-8/13*math.log(8/13,2)-5/13*math.log(5/13,2)
>>> CE
0.7350632859645522
>>> gain = EE-CE
>>> gain
0.262439260404631
```

连续与缺失值 - 缺失值处理

- 不完整样本，即样本的属性值缺失
- 仅使用无缺失的样本进行学习？

对数据信息极大的浪费

- 使用有缺失值的样本，需要解决哪些问题？

Q1：如何在属性缺失的情况下进行划分属性选择？

Q2：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？

连续与缺失值 - 缺失值处理

- $\tilde{D}$ 表示 D 中在属性 a 上没有缺失值的样本子集，$\tilde{D}^v$ 表示 $\tilde{D}$ 中在属性 a 上取值为 a^v 的样本子集，$\tilde{D}_k$ 表示 $\tilde{D}$ 中属于第 k 类的样本子集

  为每个样本 x 赋予一个权重 w_x，并定义：

  - 无缺失值样本所占的比例

$$\rho = \frac{\sum_{x \in \tilde{D}} w_x}{\sum_{x \in D} w_x}$$

  - 无缺失值样本中第 k 类所占比例

$$\tilde{p}_k = \frac{\sum_{x \in \tilde{D}_k} w_x}{\sum_{x \in \tilde{D}} w_x} \quad (1 \leq k \leq |\mathcal{Y}|)$$

  - 无缺失值样本中在属性 a 上取值 a^v 的样本所占比例

$$\tilde{r}_v = \frac{\sum_{x \in \tilde{D}^v} w_x}{\sum_{x \in \tilde{D}} w_x} \quad (1 \leq v \leq V)$$

Q1：如何在属性缺失的情况下进行划分属性选择？

连续与缺失值 - 缺失值处理

- 基于上述定义，可得

$$\begin{aligned}\mathrm{Gain}(D,a) &= \rho \times \mathrm{Gain}(\tilde{D},a) \\ &= \rho \times \left(\mathrm{Ent}(\tilde{D}) - \sum_{v=1}^{V} \tilde{r}_v \mathrm{Ent}(\tilde{D}^v)\right)\end{aligned}$$

其中

$$\mathrm{Ent}(\tilde{D}) = -\sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k \log_2 \tilde{p}_k$$

- 对于Q2
  - 若样本 x 在划分属性 a 上的取值已知，则将 x 划入与其取值对应的子结点，且样本权值在子结点中保持为 w_x
  - 若样本 x 在划分属性 a 上的取值未知，则将 x 同时划入所有子结点，且样本权值在与属性值 a^v 对应的子结点中调整为 $\tilde{r}_v \cdot w_x$ （直观来看，相当于让同一个样本以不同概率划入不同的子结点中去）

连续与缺失值 - 缺失值处理

缺失值处理实例

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

- **学习开始时，根结点包含样本集 D中全部17个样例，各样例的权值均为1**
- **以属性“色泽”为例，该属性上无缺失值的样例子集 $\tilde{D}$ 包含14个样例，$\tilde{D}$ 的信息熵为**

$$\mathrm{Ent}(\tilde{D}) = -\sum_{k=1}^{2} \tilde{p}_k \log_2 \tilde{p}_k$$

$$= -\left(\frac{6}{14}\log_2\frac{6}{14} + \frac{8}{14}\log_2\frac{8}{14}\right) = 0.985$$

连续与缺失值 - 缺失值处理

- 令 $\tilde{D}^1$，$\tilde{D}^2$，$\tilde{D}^3$ 分别表示在属性“色泽”上取值为“青绿”“乌黑”以及“浅白”的样本子集，有

$$\mathrm{Ent}(\tilde{D}^1) = -\left(\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right) = 1.000 \quad \mathrm{Ent}(\tilde{D}^2) = -\left(\frac{4}{6}\log_2\frac{4}{6} + \frac{2}{6}\log_2\frac{2}{6}\right) = 0.918$$

$$\mathrm{Ent}(\tilde{D}^3) = -\left(\frac{0}{4}\log_2\frac{0}{4} + \frac{4}{4}\log_2\frac{4}{4}\right) = 0.000$$

- 因此，样本子集 $\tilde{D}$ 上属性“色泽”的信息增益为

$$\begin{aligned}\mathrm{Gain}(\tilde{D}, \text{色泽}) &= \mathrm{Ent}(\tilde{D}) - \sum_{v=1}^{3}\tilde{r}_v\mathrm{Ent}(\tilde{D}^v) \\ &= 0.985 - \left(\frac{4}{14}\times 1.000 + \frac{6}{14}\times 0.918 + \frac{4}{14}\times 0.000\right) \\ &= 0.306\end{aligned}$$

- 于是，样本集 D 上属性“色泽”的信息增益为

$$\mathrm{Gain}(D, \text{色泽}) = \rho \times \mathrm{Gain}(\tilde{D}, \text{色泽}) = \frac{14}{17}\times 0.306 = 0.252$$

连续与缺失值 - 缺失值处理

- 类似地可计算出所有属性在数据集上的信息增益

$$\text{Gain}(D, \text{色泽}) = 0.252 \qquad \text{Gain}(D, \text{根蒂}) = 0.171$$
$$\text{Gain}(D, \text{敲声}) = 0.145 \qquad \text{Gain}(D, \text{纹理}) = 0.424$$
$$\text{Gain}(D, \text{脐部}) = 0.289 \qquad \text{Gain}(D, \text{触感}) = 0.006$$

进入“纹理=清晰”分支

进入“纹理=稍糊”分支

进入“纹理=模糊”分支

样本权重在各子结点仍为1

在属性“纹理”上出现缺失值，样本8和10同时进入3个分支，调整8和10在3分支权值分别为7/15，5/15，3/15

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

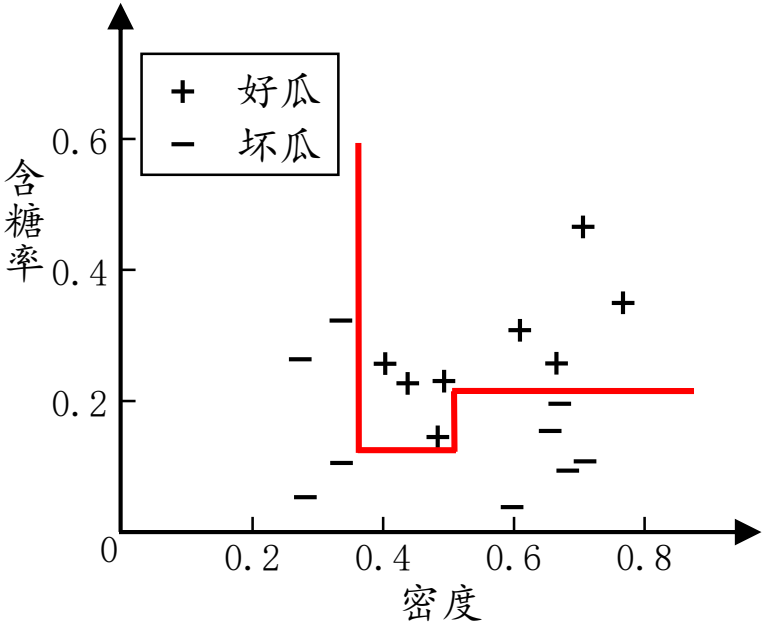
西瓜数据集2.0a上基于信息增益生成的决策树

大纲

- 基本流程
- 划分选择
- 剪枝处理
- 连续与缺失值
- **多变量决策树**

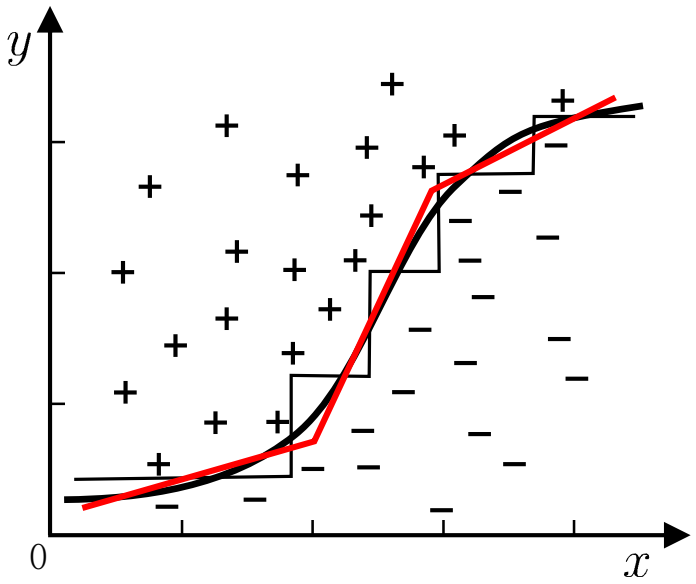
多变量决策树

- 单变量决策树分类边界：轴平行，即分类边界由若干个与坐标轴平行的分段组成。这样的分类边界使得学习结果具有较好的解释性。因为每个分段都对应了属性的某个取值。
- 当真实的学习任务比较复杂时，必须使用很多段划分才能获得较好的近视。决策树相当复杂，需要进行大量的属性测试，预测时间开销会很大。
- 若能使用斜的划分边界，则决策树模型将大为简化。多变量决策树能够实现复杂的划分。



多变量决策树

多变量决策树



- 非叶节点不再是仅对某个属性，而是对属性的线性组合

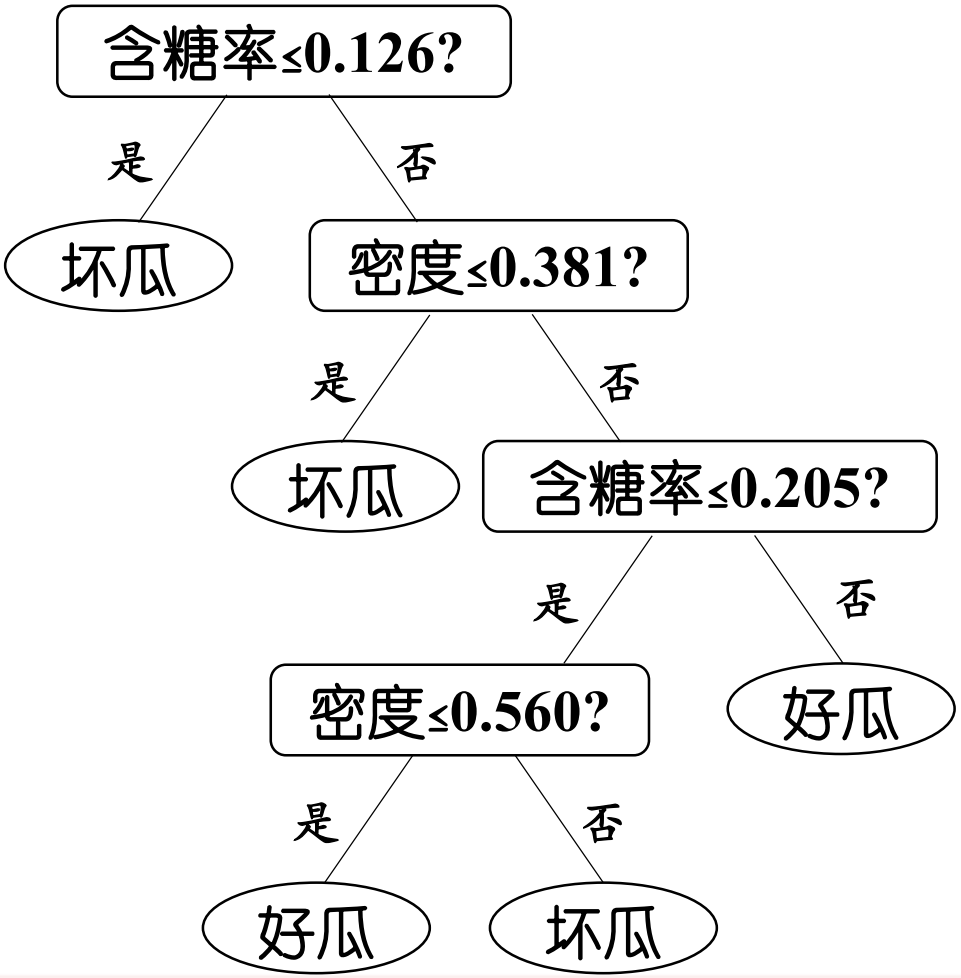
- 每个非叶结点是一个形如 $\sum_{i=1}^{d} w_i a_i = t$ 的线性分类器，其中w_i是属性a_i的权值，w_i和 t 可在该结点所含的样本集和属性集上学得

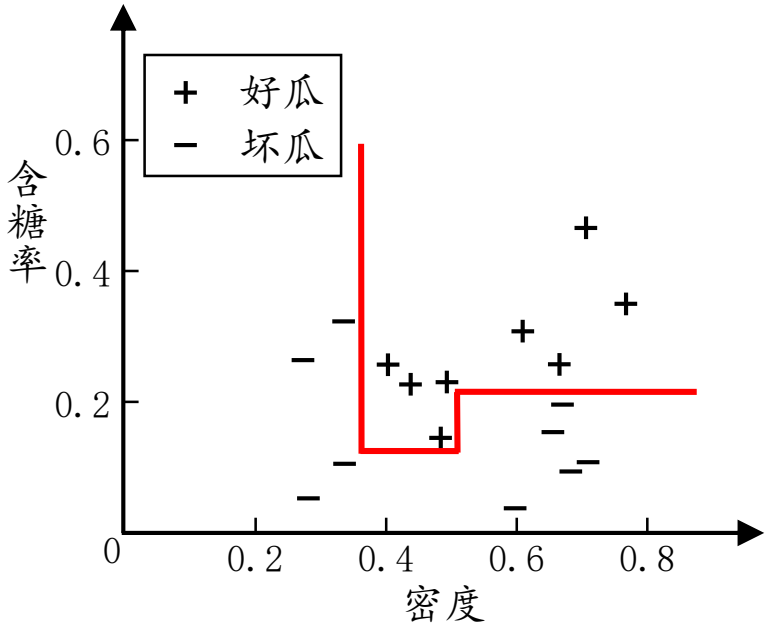
表 4.5 西瓜数据集 3.0α

编号	密度	含糖率	好瓜
1	0.697	0.460	是
2	0.774	0.376	是
3	0.634	0.264	是
4	0.608	0.318	是
5	0.556	0.215	是
6	0.403	0.237	是
7	0.481	0.149	是
8	0.437	0.211	是
9	0.666	0.091	否
10	0.243	0.267	否
11	0.245	0.057	否
12	0.343	0.099	否
13	0.639	0.161	否
14	0.657	0.198	否
15	0.360	0.370	否
16	0.593	0.042	否
17	0.719	0.103	否

多变量决策树

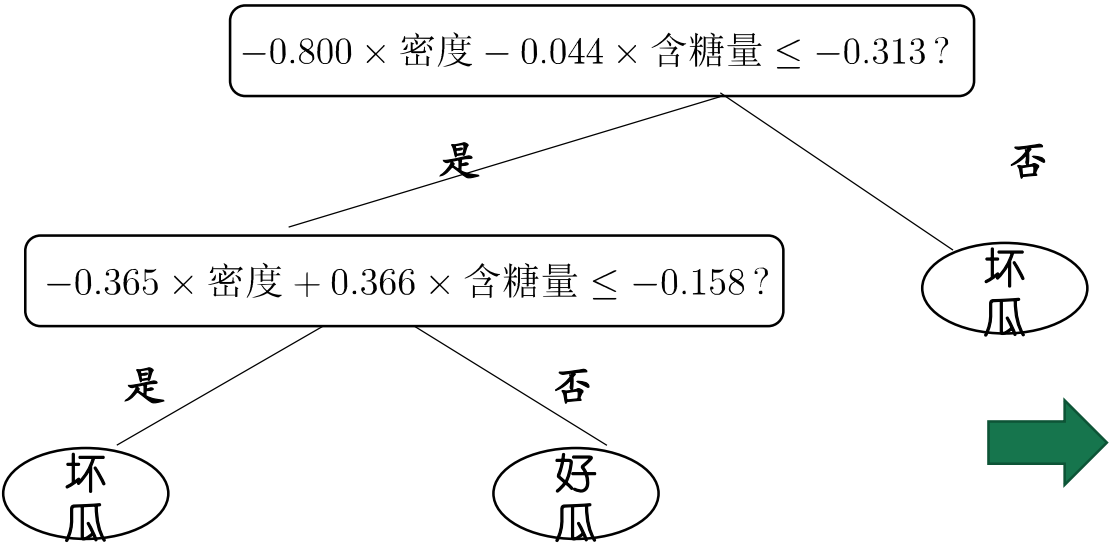
单变量决策树

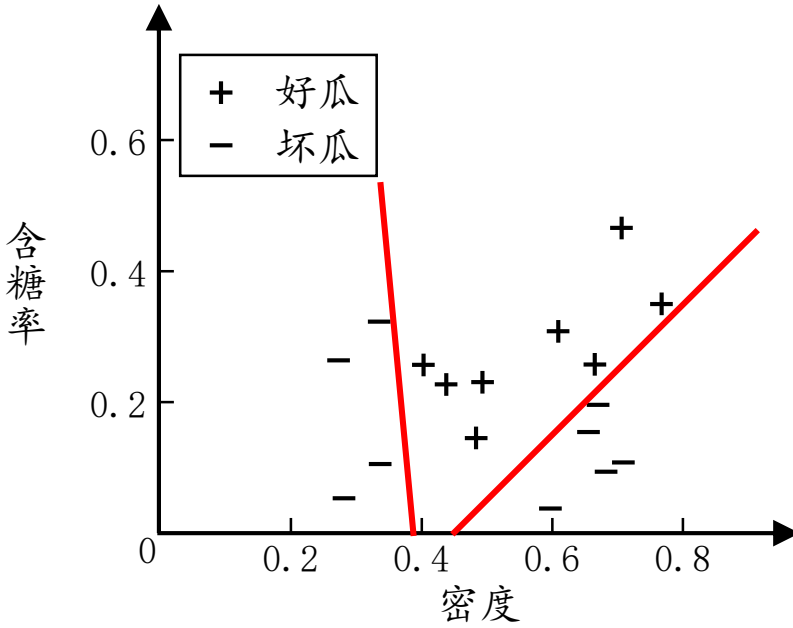




多变量决策树

多变量决策树





多变量决策树的分类边界

Take Home Message

- 属性划分选择
- 剪枝处理（预剪枝，后剪枝）
- 属性连续值和缺失值的处理
- 单变量决策树到多变量决策树

Homework

- 理解**DecisionTreeRegressor**的原理，并编程实践。