第5章 决策树

CONTENT

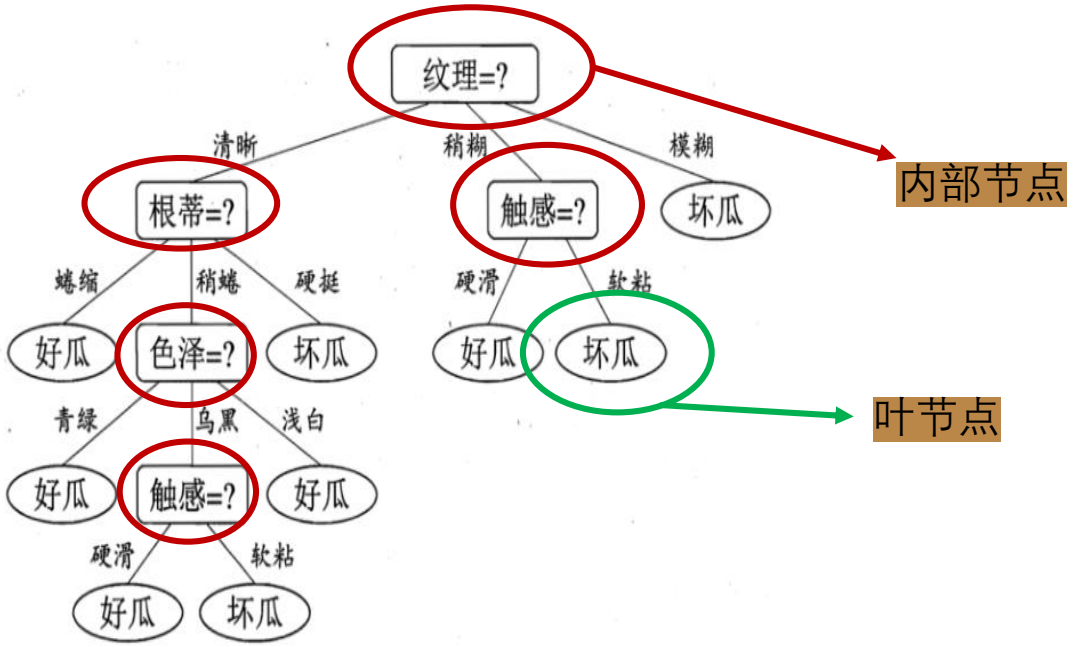
PART 01

决策树模型与学习

决策树

决策树（decision tree）：是一种基本的分类与回归方法，此处主要讨论分类的决策树。
在分类问题中，表示基于特征对实例进行分类的过程，可以认为是if-then的集合，也可以认为是定义在特征空间与类空间上的条件概率分布。

- 决策树
  - 结点
    - 内部节点：表示特征或属性
    - 叶节点：表示一个类
  - 有向边



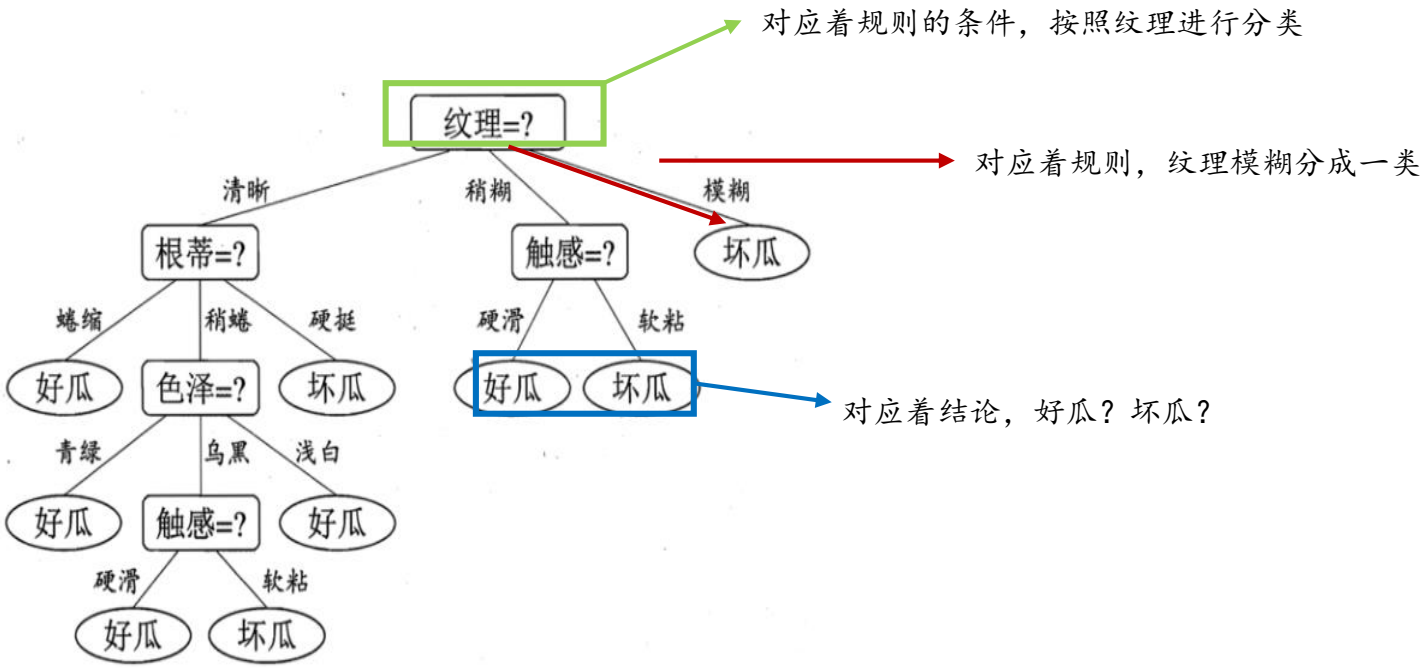
决策树

决策树与if-then规则

可以将决策树看成一个if-then规则的集合。将决策树转换成if-then规则的过程是这样的：

1. 由决策树的根结点到叶结点的每一条路径构建一条规则；
2. 路径上内部结点的特征对应着规则的条件，而叶结点的类对应着规则的结论。

决策树的路径或其对应的if-then规则集合具有一个重要的性质：互斥并且完备。这就是说，每一个实例都被一条路径或一条规则所覆盖，而且只被一条路径或一条规则所覆盖。这里所谓覆盖是指实例的特征与路径上的特征一致或实例满足规则的条件。

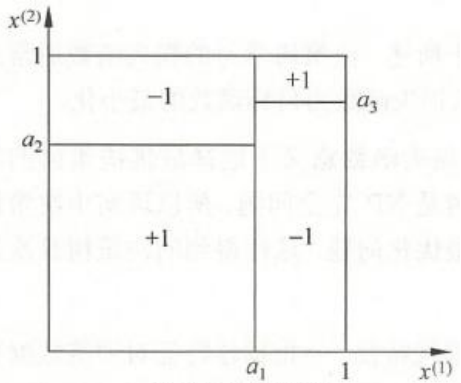


决策树

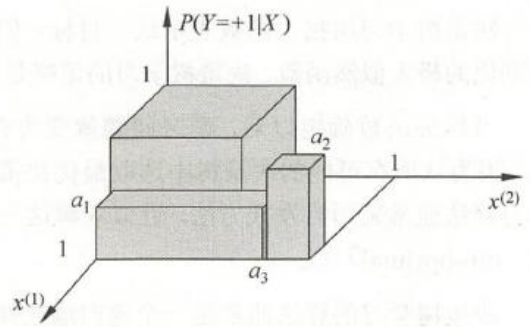
决策树与条件概率分布

决策树将特征空间划分为互不相交的单元，并在每个单元定义一个类的概率分布。

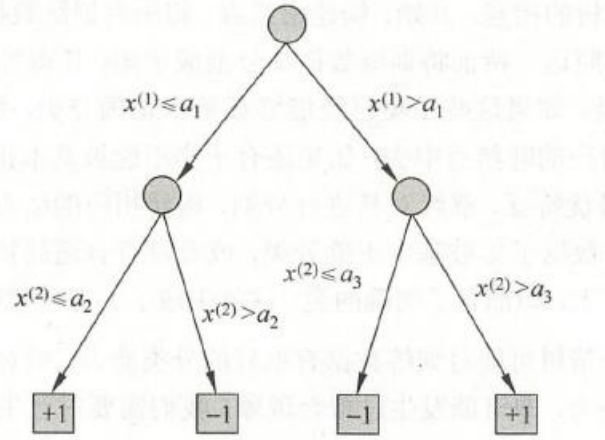
决策树的一条路径对应于划分中的一个单元，决策树所表示的条件概率分布由各个单元给定条件下类的条件概率分布组成，即P(Y|X)，叶结点（单元）上的条件概率往往偏向某一类。



(a) 特征空间划分



(b) 条件概率分布



(c) 决策树

图 5.2　决策树对应于条件概率分布

决策树

决策树学习

- 决策树学习的目标：根据给定的训练数据集构建一个决策树模型，使它能够对实例进行正确的分类。
- 决策树学习的本质：从训练数据集中归纳出一组分类规则，或者说是由训练数据集估计条件概率模型。
- 决策树学习的损失函数：正则化的极大似然函数
- 决策树学习的策略：以损失函数为目标函数的最小化
- 决策树学习的目标：在损失函数的意义下，选择最优决策树的问题

决策树

决策树学习的算法

通常是一个递归地选择最优特征，并根据该特征对训练数据进行分割，使得各个子数据集有一个最好的分类的过程。这一过程对应着对特征空间的划分，也对应着决策树的构建。

1） 开始：构建根节点，将所有训练数据都放在根节点，选择一个最优特征，按着这一特征将训练数据集分割成子集，使得各个子集有一个在当前条件下最好的分类。

2） 如果这些子集已经能够被基本正确分类，那么构建叶节点，并将这些子集分到所对应的叶节点去。

3）如果还有子集不能够被正确的分类，那么就对这些子集选择新的最优特征，继续对其进行分割，构建相应的节点，如果递归进行，直至所有训练数据子集被基本正确的分类，或者没有合适的特征为止。

4）每个子集都被分到叶节点上，即都有了明确的类，这样就生成了一颗决策树

PART 02

特征选择

决策树

特征选择：

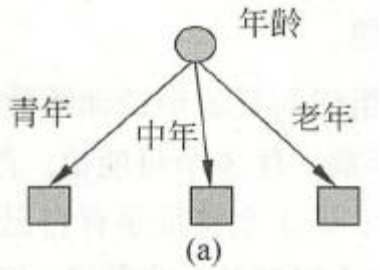
特征选择在于选取对训练数据具有分类能力的特征。

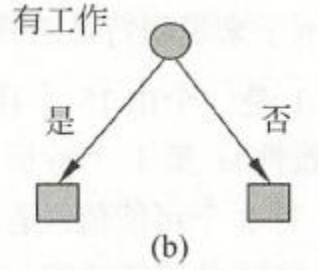
如果利用一个特征进行分类的结果与随机分类的结果没有很大区别，则称这个特征是没有分类能力的。

通常，特征选择的准则是信息增益或信息增益比。

表 5.1　贷款申请样本数据表

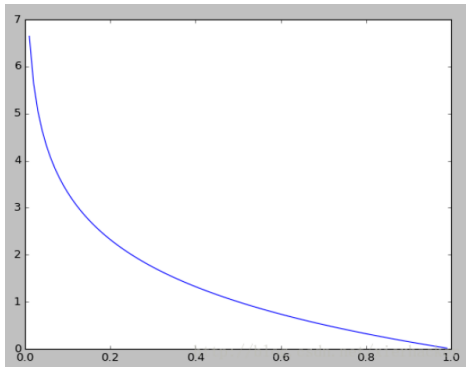
ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否





决策树

信息量$I(x) = -\log_2 p(x)$



熵（entropy）：熵度量的是随机变量的不确定性。熵越大，不确定性越大。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：$H(x) = -\sum_{i=1}^{n} P_i \log P_i$

$p_i = 0$时，$0 \log 0 = 0$

通常对数以2或e为底，则熵的单位为比特（bit）或纳特（nat）。熵只依赖于X的分布，而与X的取值无关，所以H(x)可记作H(p)

熵越大，随机变量的不确定性就越大　　$0 \le H(p) \le \log n$

$$H(p) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \cdots - -p_n \log_2 p_n = n \times (-\frac{1}{n} \log_2 \frac{1}{n}) = log_2 n$$

决策树

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，例如1,0时，即X的分布为：

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

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

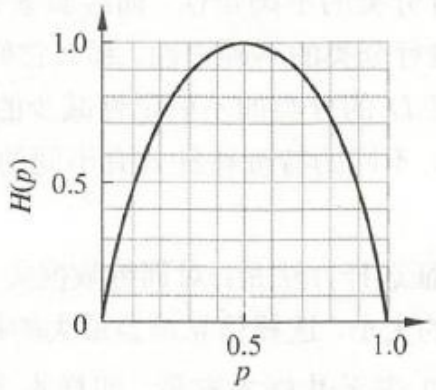


图 5.4　分布为伯努利分布时熵与概率的关系

当$p=0$或$p=1$ H(p) = 0随机变量完全没有不确定性。

当$p=0.5$ H(p) = 1熵最大，不确定性最大。

决策树

当熵中的概率由数据估计(特别是最大似然估计)得到时，所对应的熵称为经验熵。

我们定义贷款申请样本数据表中的数据为训练数据集D，则训练数据集D的经验熵为H(D)，|D|表示其样本容量，及样本个数。设有K个类Ck，
k = 1,2,3,…,K，|Ck|为属于类Ck的样本个数，经验熵公式可以写为：

$$H(x) = -\sum_{i=1}^{n} P_i \log P_i \quad \Rightarrow \quad H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|}$$

表 5.1　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

$$H(D) = -\frac{9}{15} log_2 \frac{9}{15} - \frac{6}{15} log_2 \frac{6}{15} = 0.971$$

决策树

设有随机变量(X,Y),其联合概率分布为

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

条件熵：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。随机变量X给定的条件下随机变量Y的条件熵(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：

$$H(Y|X)=\sum_{i=1}^{n}p_iH(Y|X=x_i)$$

当条件熵中的概率由数据估计（特别是极大似然估计）得到时，所对应的为经验条件熵，此时如果有0概率，令$0\log 0=0$

我们定义|Ck|为类Ck的样本个数，|D|表示样本容量。
特征A有n个不同的取值{a1,a2,...,an}，根据取值将D划分为n个子集D1,D2,...，|Di|为Di的样本个数。$D_{ik}=Di\cap C_{\mathrm{k}}$

特征A对数据集D的经验条件熵H(D|A)

$$H(Y|X)=\sum_{i=1}^{n}p_iH(Y|X=x_i)\quad\Rightarrow\quad H(D\mid A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}\log_2\frac{|D_{ik}|}{|D_i|}$$

决策树

我们定义|Ck|为类Ck的样本个数，|D|表示样本容量。
特征A有n个不同的取值{a1,a2,...,an}，根据取值将D划分为n个子集D1,D2,...，|Di|为Di的样本个数。$D_{ik} = Di \cap C_k$

特征A对数据集D的经验条件熵H(D|A)

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

表 5.1　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

$$H(D|A) = \frac{5}{15}H(D_1) + \frac{5}{15}H(D_2) + \frac{5}{15}H(D_3)$$

$$= \frac{5}{15}\left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right) + \frac{5}{15}\left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right) + \frac{5}{15}\left(-\frac{4}{5}\log_2\frac{4}{5} - \frac{1}{5}\log_2\frac{1}{5}\right)$$

$$= 0.888$$

决策树

信息增益：信息增益是相对于特征而言的。所以，特征A对训练数据集D的信息增益g(D,A)，定义为集合D的经验熵H(D)与特征A给定条件下D的经验条件熵H(D|A)之差，即：

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

H(D)表示对数据集D进行分类的不确定性。而经验条件熵H(D|A)表示在特征A给定的条件下对数据集D进行分类的不确定性。那么它们的差，即信息增益，就表示由于特征A而使得对数据集D的分类的不确定性减少的程度。显然，对于数据集D而言，信息增益依赖于特征，不同的特征往往具有不同的信息增益。信息增益大的特征具有更强的分类能力。

决策树

设训练数据集为D，|D|表示其样本容量，即样本个数。设有K个类Ck，k=1,2,.,K，|Ck|为属于类Ck的样本个数。设特征A有n个不同的取值{a1，a2,...，an},根据特征A的取值将D划分为n个子集D1，D2,...，Dn，$|Di|$为Di的样本个数，$\sum_{i=1}^{n}|Di| = |Di|$ 。 记子集Di中属于类Ck的样本的集合为Dik,即$Dik= Di \cap Ck$，$|Dik|$为Dik的样本个数。于是信息增益的算法如下。

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x信息增益

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

决策树

根据信息增益准则选择最优特征

表 5.1　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

第一步，计算经验熵H(D)

$$H(D) = -\frac{9}{15}\log_2\frac{9}{15} - \frac{6}{15}\log_2\frac{6}{15} = 0.971$$

第二步，计算各特征对数据集D的信息增益。分别以A1，A2，A3，A4表示年龄、有工作、有自己的房子和信贷情况4个特征，则

(1)

$$g(D, A_1) = H(D) - \left[\frac{5}{15}H(D_1) + \frac{5}{15}H(D_2) + \frac{5}{15}H(D_3)\right]$$

$$= 0.971 - \left[\frac{5}{15}\left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right) + \right.$$

$$\left.\frac{5}{15}\left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right) + \frac{5}{15}\left(-\frac{4}{5}\log_2\frac{4}{5} - \frac{1}{5}\log_2\frac{1}{5}\right)\right]$$

$$= 0.971 - 0.888 = 0.083$$

决策树

根据信息增益准则选择最优特征

表 5.1　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

(2)

$$g(D,A_2)=H(D)-\left[\frac{5}{15}H(D_1)+\frac{10}{15}H(D_2)\right]$$

$$=0.971-\left[\frac{5}{15}\times 0+\frac{10}{15}\left(-\frac{4}{10}\log_2\frac{4}{10}-\frac{6}{10}\log_2\frac{6}{10}\right)\right]=0.324$$

(3)

$$g(D,A_3)=0.971-\left[\frac{6}{15}\times 0+\frac{9}{15}\left(-\frac{3}{9}\log_2\frac{3}{9}-\frac{6}{9}\log_2\frac{6}{9}\right)\right]$$

$$=0.971-0.551=0.420$$

(4)

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

A_3(有自己的房子)的信息增益最大，选择特征A_3为最优特征

决策树

以信息增益作为划分训练数据集的特征，存在偏向于选择取值较多的特征的问题。使用信息增益比(information gain ratio)可以对这一问题进行校正。这是特征选择的另一准则。

信息增益比： 特征A对训练数据集D的信息增益比$g_R(D,A)$定义为其信息增益$g(D, A)$与训练数据集D关于特征A的值的熵$H_A(D)$之比，即

$$g_R(D,A)=\frac{g(D,A)}{H_A(D)} \qquad H_A(D)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\log_2\frac{|D_i|}{|D|},$$

表 5.1　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

$$H_A(D)=-\frac{5}{15}\times log_2\times\frac{5}{15}-\frac{5}{15}\times log_2\times\frac{5}{15}-\frac{5}{15}\times log_2\times\frac{5}{15}$$

PART 03

决策树的生成

决策树

ID3算法的核心是在决策树各个结点上对应信息增益准则选择特征，递归地构建决策树。具体方法是：

1）从根结点(root node)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。

2）由该特征的不同取值建立子节点，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；

3）最后得到一个决策树。

决策树

输入:训练数据集D,特征集A阈值 ε ;

输出:决策树T。

(1)若D中所有实例属于同一类C_k,则T为单结点树，并将类C_k作为该结点的类标记，返回T;

(2)若A=0,则T为单结点树，并将D中实例数最大的类C_k作为该结点的类标记，返回T;

(3)否则，按算法5.1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_g;

(4)如果A_g的信息增益小于阈值 ε ,则置T为单结点树，并将D中实例数最大的类C_k作为该结点的类标记，返回T;

(5)否则，对A_g的每一可能值a_i，依$A_g = a_i$,将D分割为若干非空子集D_i,将D_i中实例数最大的类作为标记，构建子结点，由结点及其子结点构成树T,返回T;

(6)对第i个子结点，以D_i为训练集，以$A-\{A_g\}$为特征集，递归地调用步(1)-步(5),得到子树T_i,返回T_i

决策树

利用ID3算法建立决策树

表 5.1　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

由于特征A3（有自己的房子）的信息增益值最大，所以选择特征A3作为根结点的特征。它将训练数据集D划分为两个子集D1（A3取值为“是"）和D2（A3取值为“否”）。由于D1只有同一类的样本点，所以它成为一个叶结点，结点的类标记为“是”。对D2则需从特征A1（年龄），A2（有工作）和A4（信贷情况）中选择新的特征。计算各个特征的信息增益：

$$g(D_2, A_1) = H(D_2) - H(D_2|A_1)$$

$$H(D_2) = -\frac{6}{9} \times log_2 \times \frac{6}{9} - \frac{3}{9} \times log_2 \times \frac{3}{9} = 0.918$$

$$H(D_2|A_1) = \frac{4}{9} \times (-\frac{3}{4} \times log_2 \times \frac{3}{4} - \frac{1}{4} \times log_2 \times \frac{1}{4}) + \frac{2}{9} \times (-\frac{2}{2} \times log_2 \times \frac{2}{2} - 0 \times log_2 \times 0) + \frac{3}{9} \times (-\frac{1}{3} \times log_2 \times \frac{1}{3} - \frac{2}{3} \times log_2 \times \frac{2}{3}) = 0.667$$

$$g(D_2, A_1) = H(D_2) - H(D_2|A_1) = 0.918 - 0.667 = 0.251$$

类似的，对其它特征进行计算

$$g(D_2, A_2) = H(D_2) - H(D_2|A_2) = 0.918$$

$$g(D_2, A_4) = H(D_2) - H(D_2|A_4) = 0.474$$

决策树

利用ID3算法建立决策树

表 5.1　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

$$g(D_2, A_1) = H(D_2) - H(D_2|A_1) = 0.918 - 0.667 = 0.251$$

$$g(D_2, A_2) = H(D_2) - H(D_2|A_2) = 0.918$$

$$g(D_2, A_4) = H(D_2) - H(D_2|A_4) = 0.474$$

选择信息增益最大的特征A2（有工作）作为结点的特征。由于A2有两个可能取值，从这一结点引出两个子结点：一个对应“是”（有工作）的子结点，包含3个样本，它们属于同一类，所以这是一个叶结点，类标记为“是”；另一个是对应“否”（无工作）的子结点，包含6个样本，它们也属于同一类，所以这也是一个叶结点，类标记为“否”。

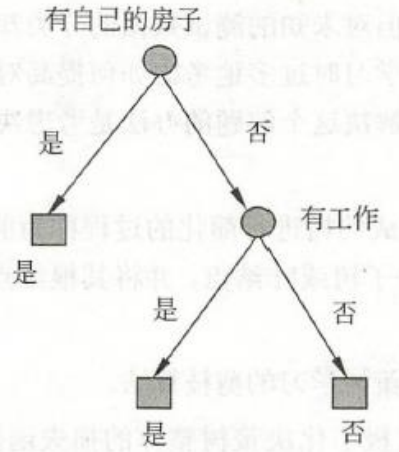


图 5.5　决策树的生成

决策树

利用ID3算法建立决策树

```
from sklearn import tree
import graphviz

X = [[1, 2, 2, 3], [1, 2, 2, 2], [1, 1, 2, 2], [1, 1, 1, 3], [1, 2, 2, 3], [2, 2, 2, 3], [2, 2, 2, 2], [2, 1, 1, 2],
     [2, 2, 1, 1], [2, 2, 1, 1],
     [3, 2, 1, 1], [3, 2, 1, 2], [3, 1, 2, 2], [3, 1, 2, 1], [3, 2, 2, 2]]
Y = [2, 2, 1, 1, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2]
clf = tree.DecisionTreeClassifier(criterion="entropy")
clf = clf.fit(X, Y)
dot_data = tree.export_graphviz(clf, out_file=None,
                                feature_names=['age', 'job', 'house', 'credit'],
                                class_names=['yes', 'no'],
                                filled=True, rounded=True,
                                special_characters=True)
graph = graphviz.Source(dot_data)
graph.render("iris")
```

决策树

利用ID3算法建立决策树

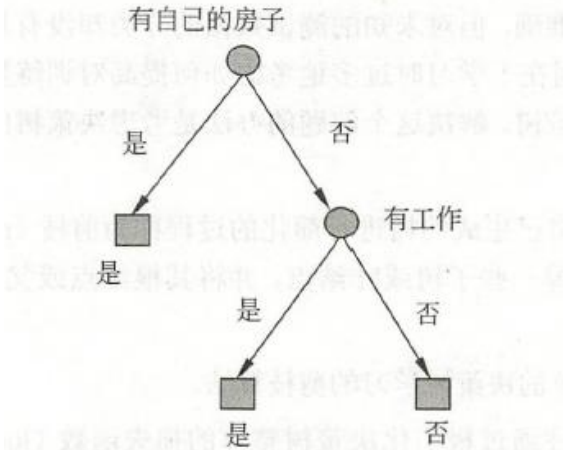
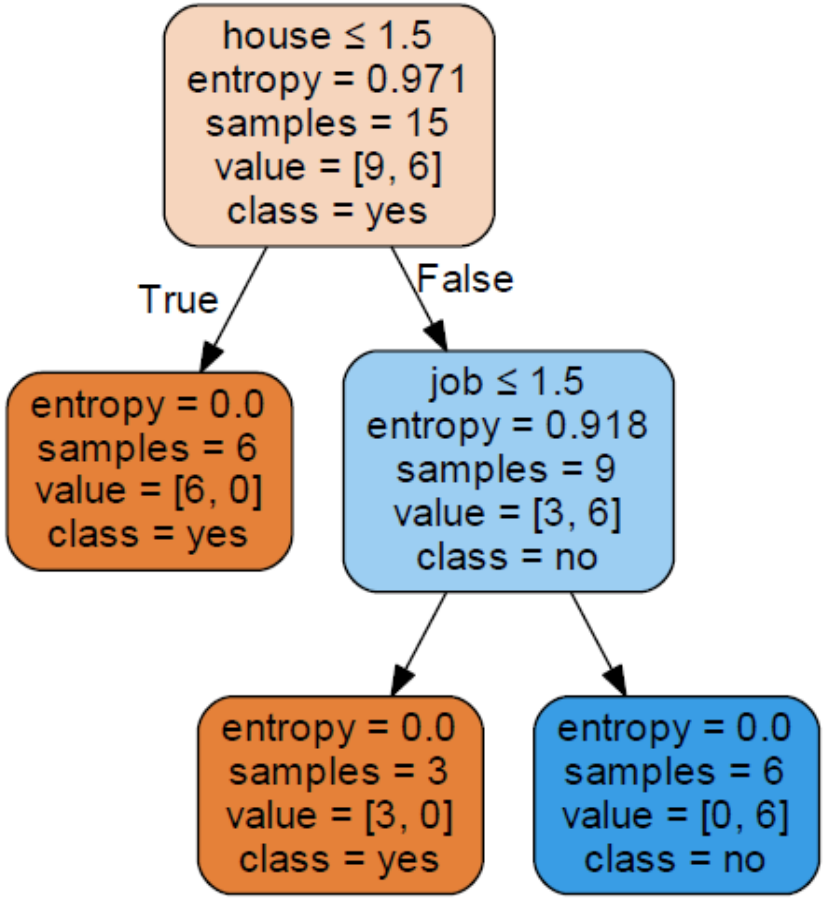


图 5.5　决策树的生成



PART 04

决策树的剪枝

剪枝处理

剪枝(pruning)是决策树学习算法对付“过拟合”的主要手段.在决策树学习中，为了尽可能正确分类训练样本,结点划分过程将不断重复，有时会造成决策树分支过多,这时就可能因训练样本学得“太好”了,以致于把训练集自身的一些特点当作所有数据都具有的一般性质而导致过拟合.因此,可通过主动去掉一些分支来降低过拟合的风险.

剪枝处理

- 预剪枝是指在决策树生成过程中,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标记为叶结点;

- 后剪枝则是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节点。

决策树

剪枝处理—预剪枝

表 4.2 西瓜数据集 2.0 划分出的训练集(双线上部)与验证集(双线下部)

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

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

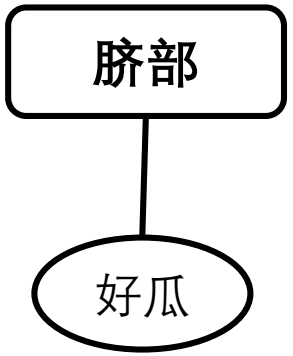
决策树

剪枝处理—预剪枝

表 4.2 西瓜数据集 2.0 划分出的训练集(双线上部)与验证集(双线下部)

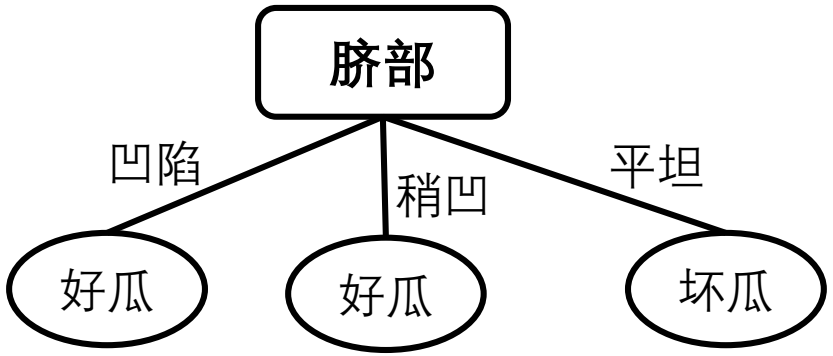
编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否
编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	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

没有划分 验证集精度为$\frac{3}{7}\times 100\%=42.9\%$



划分

有划分 验证集精度为$\frac{5}{7}\times 100\%=71.4\%$



决策树

剪枝处理—预剪枝

未剪枝的决策树

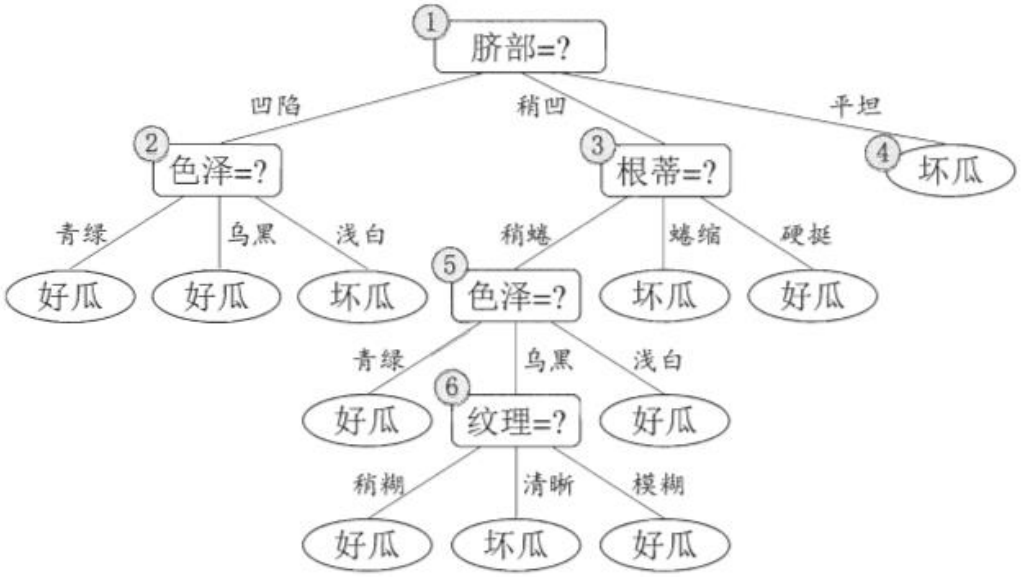


图 4.5 基于表 4.2 生成的未剪枝决策树

预剪枝的决策树

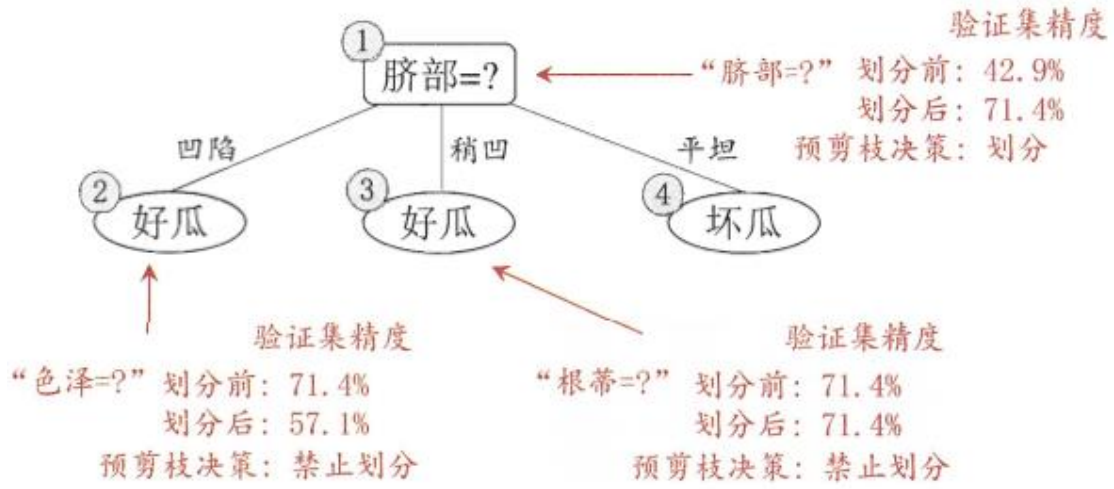


图 4.6 基于表 4.2 生成的预剪枝决策树

预剪枝使得决策树的很多分支都没有“展开”,这不仅降低了过拟合的风险，还显著减少了决策树的训练时间开销和测试时间开销.但另一方面，有些分支的当前划分虽不能提升泛化性能、甚至可能导致泛化性能暂时下降,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，给预剪枝决策树带来了欠拟合的风险.

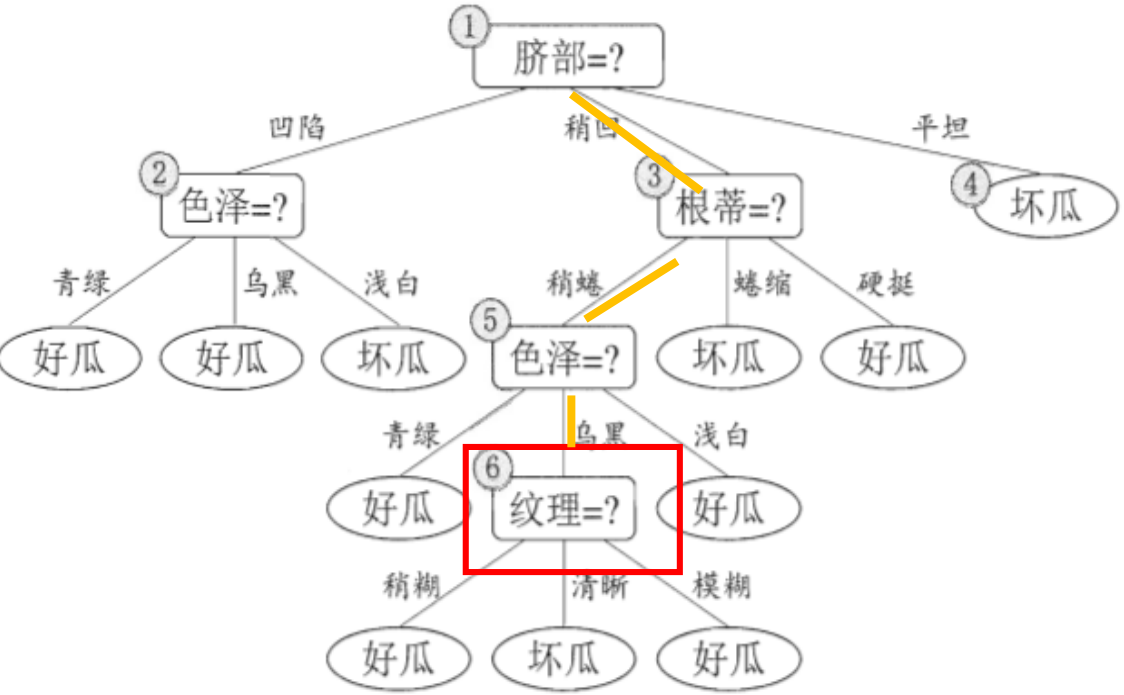
剪枝处理—后剪枝

- 后剪枝就是先构造一颗完整的决策树，然后自底向上的对非叶结点进行考察，若将该结点对应的子树换为叶结点能够带来泛华性能的提升，则把该子树替换为叶结点。

决策树

剪枝处理—后剪枝

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



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

原本验证集精度为$\frac{3}{7}\times 100\%=42.9\%$

纹理包括训练数据(7,15),设为好瓜

测试数据(4,8,11,12),分类正确验证集精度为$\frac{4}{7}\times 100\%=57.1\%$

剪枝

剪枝处理—后剪枝

未剪枝的决策树

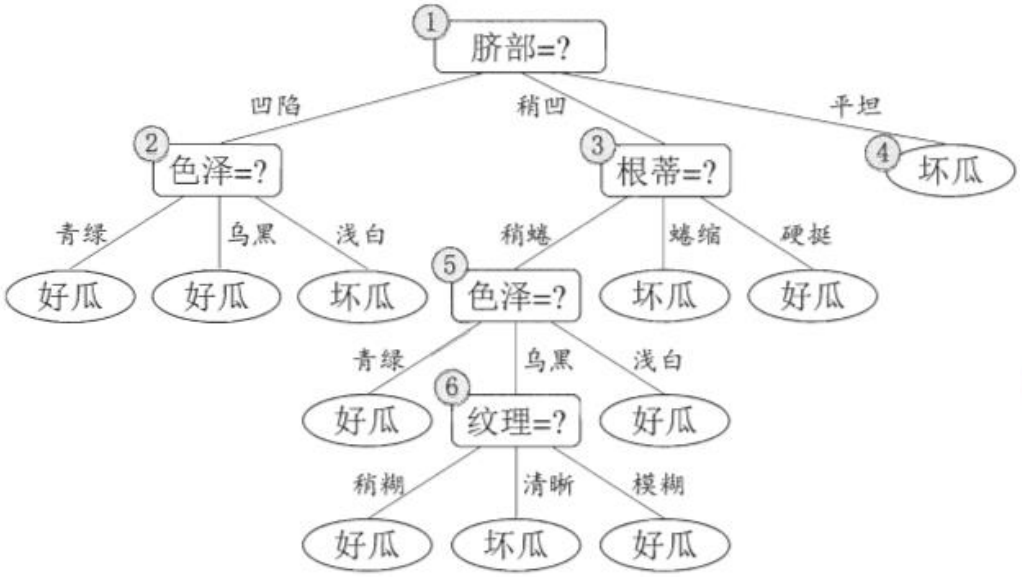


图 4.5 基于表 4.2 生成的未剪枝决策树

后剪枝的决策树

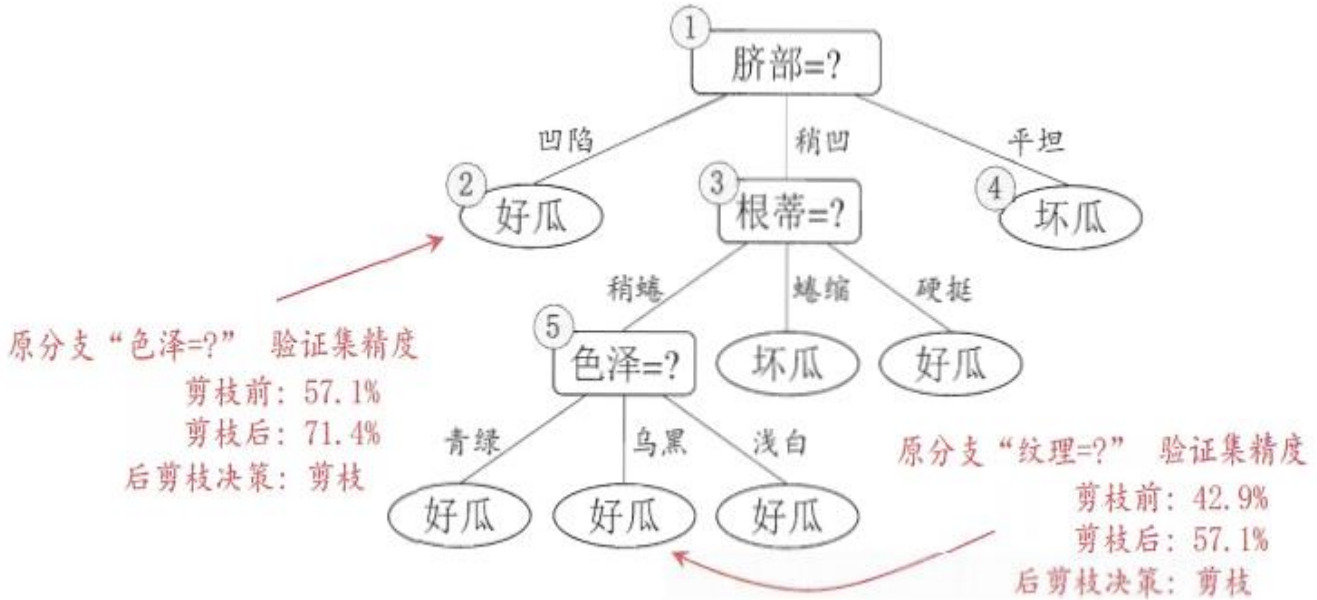


图 4.7 基于表 4.2 生成的后剪枝决策树

后剪枝决策树通常比预剪枝决策树保留了更多的分支. 一般情形下，后剪枝决策树的欠拟合风险很小，泛化性能往往优于预剪枝决策树.但后剪枝过程是在生成完全决策树之后进行的，并且要自底向上地对树中的所有非叶结点进行逐一考察,因此其训练时间开销比未剪枝决策树和预剪枝决策树都要大得多.

剪枝处理

- 决策树的剪枝往往通过极小化决策树整体的损失函数(loss function)或代价函数(cost function) 来实现。设树T的叶结点个数为|T|, t是树T的叶结点，该叶结点有 N_t 个样本点，其中k类的样本点有 N_{tk} 个，k= 1,2,⋯,K, $H_t(T)$ 为叶结点t上的经验熵，$\alpha \geqslant 0$ 为参数，则决策树学习的损失函数可以定义为

$$C_\alpha(T) = \sum_{t=1}^{|T|} N_t H_t(T) + \alpha|T|$$

其中，经验熵为

$$H_t(T) = -\sum_k \frac{N_{tk}}{N_t} \log \frac{N_{tk}}{N_t}$$

左边第一项

$$C(T) = \sum_{t=1}^{|T|} N_t H_t(T) = -\sum_{t=1}^{|T|} \sum_{k=1}^{K} N_{tk} \log \frac{N_{tk}}{N_t}$$

总的

$$C_\alpha(T) = C(T) + \alpha|T|$$

决策树

剪枝处理

$$C_\alpha(T) = \sum_{t=1}^{|T|} N_t H_t(T) + \alpha|T| \longrightarrow C_\alpha(T) = C(T) + \alpha|T|$$

- C(T)只由训练数据决定，它反映了模型对训练数据的预测误差，即模型与训练数据的拟合程度
- |T|表示模型复杂度(等比于叶节点个数)
- 参数$\alpha \geq 0$控制两者之间的平衡

a) 较大的a的情况下:|T|就被迫要尽量小，但是C(T)相对地就会增加，这动态地促使模型选择较简单的模型(浅树)

b) 较小的a的情况下: |T|就被迫要尽量大，C(T)就会相对地减小，促使选择较复杂的模型(深树)

c) $\alpha = 0$意味着只考虑模型与训练数据的拟合程度,不考虑模型的复杂度。

剪枝处理

算法 5.4（树的剪枝算法）

输入：生成算法产生的整个树 T，参数 α；

输出：修剪后的子树 T_α。

（1）计算每个结点的经验熵。

（2）递归地从树的叶结点向上回缩。

设一组叶结点回缩到其父结点之前与之后的整体树分别为 T_B 与 T_A，其对应的损失函数值分别是 $C_\alpha(T_B)$ 与 $C_\alpha(T_A)$，如果

$$C_\alpha(T_A) \leqslant C_\alpha(T_B) \tag{5.15}$$

则进行剪枝，即将父结点变为新的叶结点。

（3）返回（2），直至不能继续为止，得到损失函数最小的子树 T_α。 ■

PART 05

CART算法

CART算法

CART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。CART是在给定输入随机变量X条件下输出随机变量Y的条件概率分布的学习方法。CART假设决策树是二叉树，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。

CART算法

CART算法由以下两步组成:

(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。

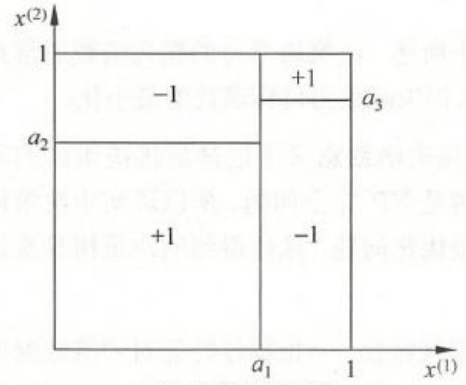
CART生成---回归树的生成

一棵回归树对应着输入空间(即特征空间)的一个划分以及在划分的单元上的输出值。假设已将输入空间划分为M个单元R1, R2,... ,RM,并且在每个单元Rm上有一个固定的输出值Cm,于是回归树模型可表示为

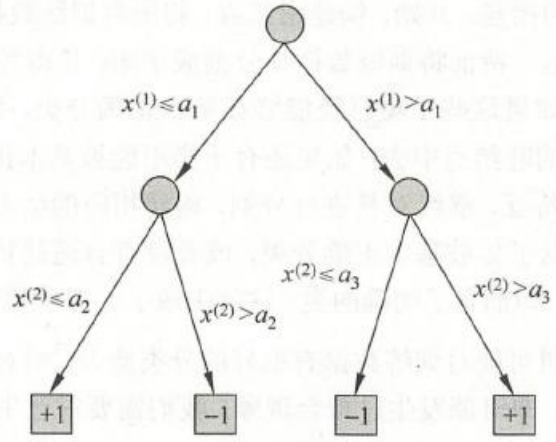
$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m)$$

预测误差

$$\sum_{x_i \in R_m} (y_i - f(x_i))^2$$



(a) 特征空间划分



(c) 决策树

图 5.2　决策树对应于条件概率分布

决策树

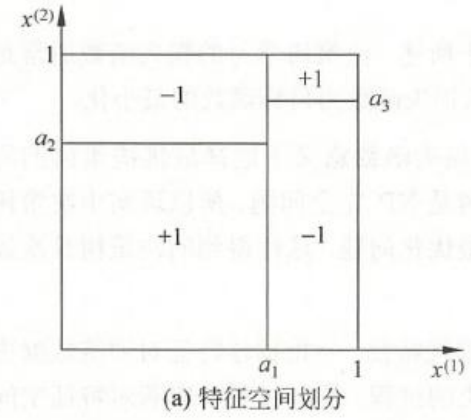
CART生成---回归树的生成

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

预测误差

$$\sum_{x_i \in R_m} (y_i - f(x_i))^2$$

平方误差最小化准则



(a) 特征空间划分

给定一个随机数列$\{(x_1,y_1),(x_2,y_2),\dots,(x_n,y_n)\}$　设该空间中最优的输出值为a.
根据最小平方误差准则，构造a的函数如下:
$F(\mathrm{a})=(y_1-a)^2+(y_2-a)^2+\cdots+(y_n-a)^2$
考察其单调性，
$F'(\mathrm{a})=-2(y_1-a)-2(y_2-a)+\cdots-2(y_n-a)=2\mathrm{an}-2\sum_{i=1}^{n} y_i$

令$F'(\mathrm{a})$ =0得，　$2\mathrm{an}-2\sum_{i=1}^{n} y_i$=0　　$a=\frac{\sum_{i=1}^{n} y_i}{n}$

根据其单调性，易知　当 $a=\frac{\sum_{i=1}^{n} y_i}{n}$　时 F(a)的值最小

单元R_m上的C_m的最优值$\widehat{C_m}$是R_m上的所有输入实例x_i对应的输出Y_i的均值，即

$$\hat{c}_m=\mathrm{ave}(y_i|x_i \in R_m)$$

CART生成---回归树的生成

单元R_m上的C_m的最优值$\hat{c}_m$是R_m上的所有输入实例x_i对应的输出Y_i的均值，即

$$\hat{c}_m = \text{ave}(y_i|x_i \in R_m)$$

这里采用启发式的方法，选择第j个变量$x^{(j)}$和它取的值s,作为切分变量(splitting variable)和切分点(splitting point)，并定义两个区域：

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

然后寻找最优切分变量j和最优切分点s。具体地，求解

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

$$\min_{j,s}\left[\sum_{x_i\in R_1(j,s)}(y_i-\hat{C}_1)^2+\sum_{x_i\in R_2(j,s)}(y_i-\hat{C}_2)^2\right]$$

决策树

CART生成---回归树的生成

$$\min_{j,s}\left[\sum_{x_i\in R_1(j,s)}(y_i-\hat{C}_1)^2+\sum_{x_i\in R_2(j,s)}(y_i-\hat{C}_2)^2\right]$$

$$\hat{c}_m=\text{ave}(y_i|x_i\in R_m)$$

遍历所有输入变量，找到最优的切分变量j,构成一个对(j,s)。依此将输入空间划分为两个区域。接着，对每个区域重复上述划分过程，直到满足停止条件为止。这样就生成一棵回归树。这样的回归树通常称为最小二乘回归树(least squares regression tree).

决策树

CART生成---回归树的生成

算法 5.5（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

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right] \tag{5.21}$$

遍历变量 j，对固定的切分变量 j 扫描切分点 s，选择使式 (5.21) 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：

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

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

（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 $R_1,R_2,\cdots,R_M$，生成决策树：

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

■

决策树

下表为训练数据集，特征向量只有一维，根据此数据表建立回归决策树。

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05

(1) 选择最优切分变量j与最优切分点s

在本数据集中，只有一个特征变量，最优切分变量自然是x。
接下来考虑9个切分点(切分变量两个相邻取值区间内任意一点均可)

$\{1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5\}$

当s=1.5时，根据之前的公式

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

划分成如下两个区域

$R_1 = \{1\}$

$R_2 = \{2,3,4,5,6,7,8,9,10\}$

决策树

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05

$$\hat{c}_m = \text{ave}(y_i | x_i \in R_m)$$

$$\min_{j,s}\left[\sum_{x_i \in R_1(j,s)} (y_i - \hat{C}_1)^2 + \sum_{x_i \in R_2(j,s)} (y_i - \hat{C}_2)^2\right]$$

(1) 选择最优切分变量j与最优切分点s $\{1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5\}$

当s=1.5时， $R_1 = \{1\}$ $R_2 = \{2,3,4,5,6,7,8,9,10\}$

$\hat{C}_1 = 5.56$ $\hat{C}_2 = \frac{1}{9} \times (5.7 + 5.91 + \cdots 9.05) = 7.5$

$$\sum_{x_i \in R_1(j,s)} (y_i - \hat{C}_1)^2 + \sum_{x_i \in R_2(j,s)} (y_i - \hat{C}_2)^2 = (5.56 - 5.56)^2 + (5.7 - 7.5)^2 + (5.91 - 7.5)^2 + \cdots + (9.05 - 7.5)^2$$

$$= 15.72$$

同理，计算得到其他各切分点的损失函数值，列表如下

第一次应该选择的切分点s为6.5

S	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
L(S)	15.72	12.07	8.36	5.78	3.91	1.93	8.01	11.73	15.74

决策树

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05

S	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
L(S)	15.72	12.07	8.36	5.78	3.91	1.93	8.01	11.73	15.74

(2)用选定的对(j,s)划分区域并决定相应的输出值

$\{1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5\}$

当s=6.5时，$R_1 = \{1,2,3,4,5,6\}$　　$R_2 = \{7,8,9,10\}$

$$\hat{C}_1 = \frac{1}{6} \times (5.56 + 5.7 + 5.91 + 6.4 + 6.8 + 7.05) = 6.24$$

$$\hat{C}_2 = \frac{1}{4} \times (8.9 + 8.7 + 9 + 9.05) = 8.91$$

决策树

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05

(3)继续对两个子区域调用步骤(1),(2) 直至满足停止条件

对R1区域　　取切分点 $\{1.5, 2.5, 3.5, 4.5, 5.5\}$　　计算得到单元输出值为

s	1.5	2.5	3.5	4.5	5.5
C1	5.56	5.63	5.72	5.89	6.07
C2	6.37	6.54	6.75	6.93	7.05

损失函数值为

S	1.5	2.5	3.5	4.5	5.5
L(S)	1.3087	0.754	0.2771	0.4368	1.0644

L(3.5)最小，取s=3.5为划分点。

决策树

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05

(4)将输入空间划分为M个区域$R_1, R_2, \cdots, R_m$生成决策树

假设两次划分后即停止，则最终生成的回归树为：

$$T = \begin{cases} 5.72 & x \leqslant 3.5 \\ 6.75 & 3.5 < x \leqslant 6.5 \\ 8.91 & x > 6.5 \end{cases}$$

CART生成---分类树的生成

分类树用基尼指数选择最优特征，同时决定该特征的最优二值切分点

基尼指数：分类问题中，假设有K个类，样本点属于第k类的概率为Pk,则概率分布的基尼指数定义为

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

对于二类分类问题，若样本点属于第1个类的概率是p,则概率分布的基尼指数为

$$\text{Gini}(p) = 2p(1-p)$$

对于给定的样本集合D,其基尼指数为

$$\text{Gini}(D) = 1 - \sum_{k=1}^{K} \left(\frac{|C_k|}{|D|} \right)^2$$

这里，Ck是D中属于第k类的样本子集，K是类的个数。

CART生成---分类树的生成

如果样本集合D根据特征A是否取某一可能值a被分割成D1和D2两部分，即

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

则在特征A的条件下，集合D的基尼指数

$$\text{Gini}(D, A) = \frac{|D_1|}{|D|}\text{Gini}(D_1) + \frac{|D_2|}{|D|}\text{Gini}(D_2)$$

基尼指数Gini(D)表示集合D的不确定性，基尼指数Gini(D, A)表示经A=a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。

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

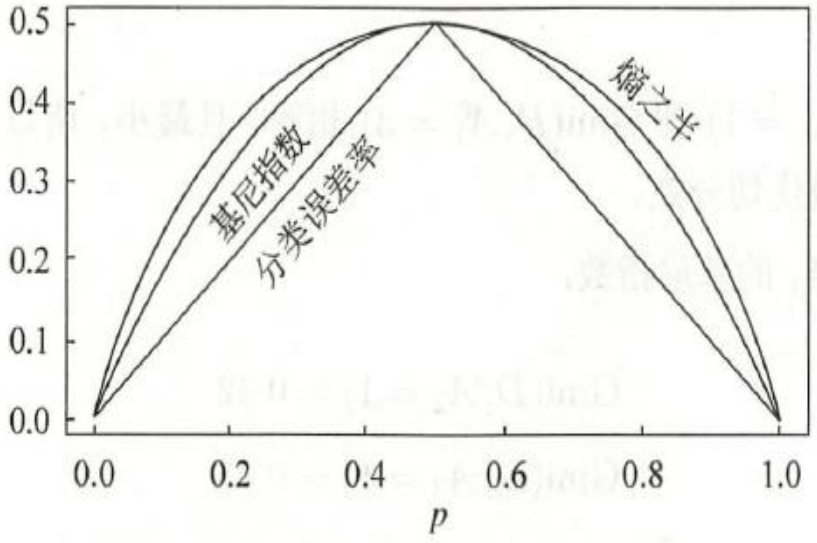


图 5.7　二类分类中基尼指数、熵之半和分类误差率的关系

$$H(x) = -\sum_{i=1}^{n} P_i \log P_i \qquad H(p) = -p\log_2 p - (1-p)\log_2(1-p)$$

CART生成---分类树的生成

CART生成算法

- 输入：训练数据集D，停止计算条件
- 输出：CART决策树

从根节点开始，递归对每个结点操作

① 设结点数据集为D，对每个特征A，对其每个值a，根据样本点对A=a的测试为是或否，将D分为D1，D2，计算A=a的基尼指数

② 在所有的特征A以及所有可能的切分点a中，选择基尼指数最小的特征和切分点，将数据集分配到两个子结点中。

③ 对两个子结点递归调用1，2步骤，直至满足停止条件

④ 生成CART树

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

决策树

CART生成---分类树的生成

应用CART算法生成决策树

表 5.1　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

① 计算各特征的基尼指数，选择最优特征以及其最优切分点。分别以A1, A2, A3, A4表示年龄、有工作、有自己的房子和信贷情况4个特征，并以1, 2, 3表示年龄的值为青年、中年和老年，以1, 2表示有工作和有自己的房子的值为是和否，以1, 2, 3表示信贷情况的值为非常好、好和一般。

求特征A1的基尼指数

$$\mathrm{Gini}(D,A)=\frac{|D_1|}{|D|}\mathrm{Gini}(D_1)+\frac{|D_2|}{|D|}\mathrm{Gini}(D_2)$$

$$Gini(D,A1=1)=\frac{5}{15}\times(2\times\frac{2}{5}\times(1-\frac{2}{5}))+\frac{10}{15}\times(2\times\frac{7}{10}\times(1-\frac{7}{10}))$$

$$=0.44$$

$$\mathrm{Gini}(D,A_1=2)=0.48$$

$$\mathrm{Gini}(D,A_1=3)=0.44$$

决策树

CART生成---分类树的生成

应用CART算法生成决策树

表 5.1　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

求特征A_2和A_3的基尼指数

$$\mathrm{Gini}(D, A_2 = 1) = 0.32$$

$$\mathrm{Gini}(D, A_3 = 1) = 0.27$$

由于A_2 A_3只有一个切分点，所以他们就是最优切分点

求特征A4的基尼指数

$$\mathrm{Gini}(D, A_4 = 1) = 0.36$$

$$\mathrm{Gini}(D, A_4 = 2) = 0.47$$

$$\mathrm{Gini}(D, A_4 = 3) = 0.32$$

A_4=3为A_4的最优切分点

A_3为最优特征，A_3= 1为其最优切分点

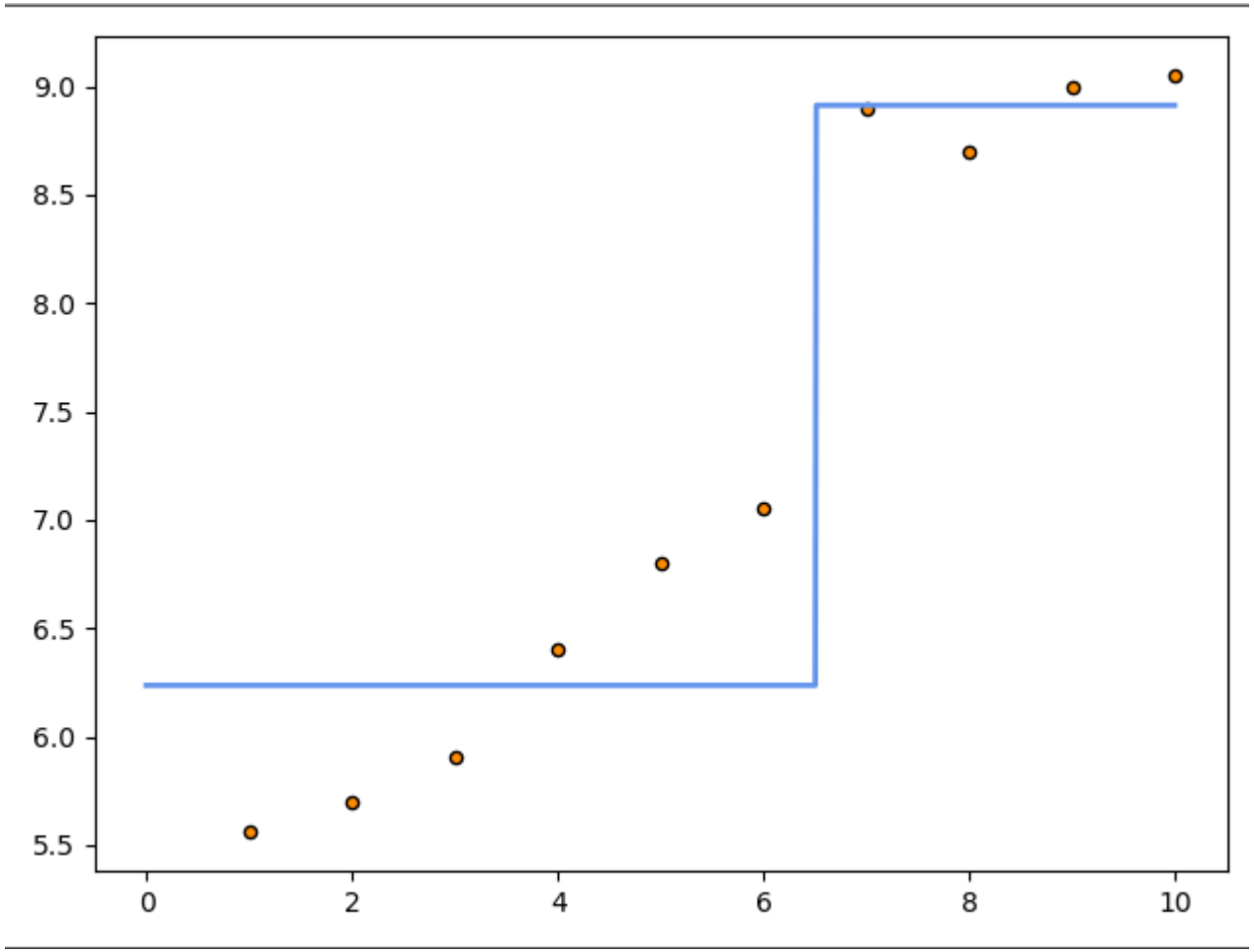
根结点生成两个子结点，一个是叶结点。

对另一个结点继续使用以上方法在A_1, A_2, A_4中选择最优特征及其最优切分点，结果是A_2= 1。依此计算得知，所得结点都是叶结点。

决策树

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn import tree

# Data set
X = np.array(list(range(1, 11))).reshape(-1, 1)
y = np.array([5.56, 5.70, 5.91, 6.40, 6.80, 7.05, 8.90, 8.70, 9.00, 9.05]).ravel()

clf = tree.DecisionTreeRegressor(max_depth=1)
clf = clf.fit(X, y)

# Predict
X_test = np.arange(0.0, 10.0, 0.01)[:, np.newaxis]
y_1 = clf.predict(X_test)

# Plot the results
plt.figure()
plt.scatter(X, y, s=20, edgecolor="black", c="darkorange", label="data")
plt.plot(X_test, y_1, color="cornflowerblue", linewidth=2)
plt.show()
```

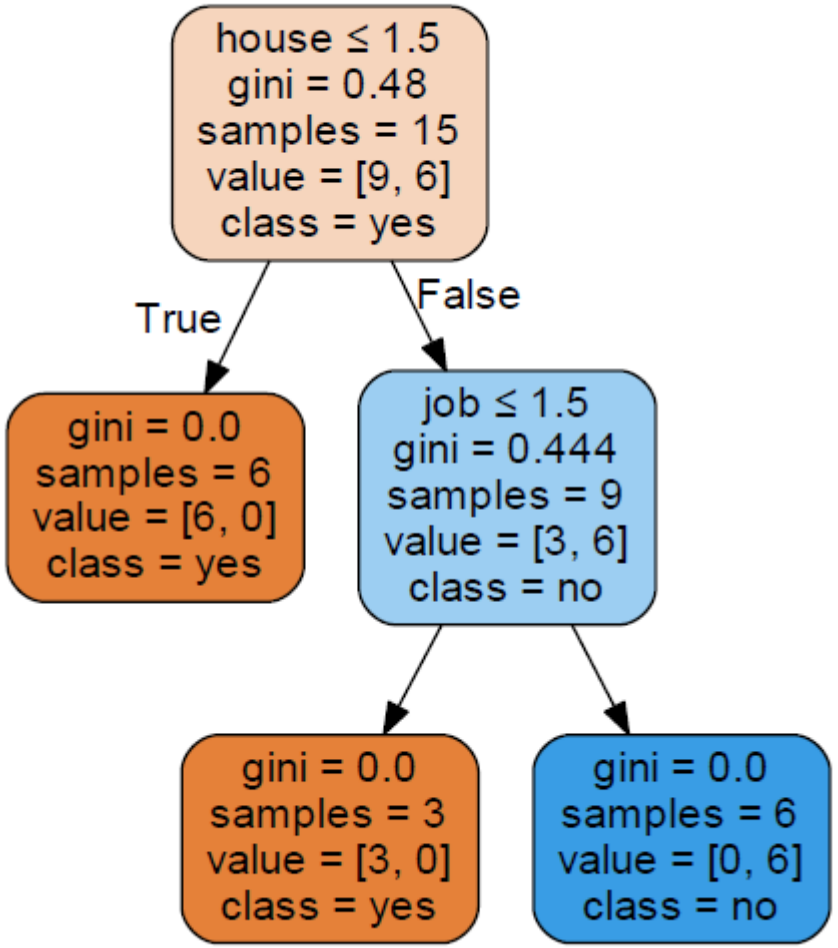
决策树



决策树

```
from sklearn import tree
import graphviz

X = [[1, 2, 2, 3], [1, 2, 2, 2], [1, 1, 2, 2], [1, 1, 1, 3], [1, 2, 2, 3], [2, 2, 2, 3], [2, 2, 2, 2], [2, 1, 1, 2],
     [2, 2, 1, 1], [2, 2, 1, 1],
     [3, 2, 1, 1], [3, 2, 1, 2], [3, 1, 2, 2], [3, 1, 2, 1], [3, 2, 2, 2]]
Y = [2, 2, 1, 1, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2]
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, Y)
dot_data = tree.export_graphviz(clf, out_file=None,
                                feature_names=['age', 'job', 'house', 'credit'],
                                class_names=['yes', 'no'],
                                filled=True, rounded=True,
                                special_characters=True)
graph = graphviz.Source(dot_data)
graph.render("classifier_iris")
```

决策树


house ≤ 1.5
gini = 0.48
samples = 15
value = [9, 6]
class = yes
True
False
gini = 0.0
samples = 6
value = [6, 0]
class = yes
job ≤ 1.5
gini = 0.444
samples = 9
value = [3, 6]
class = no
gini = 0.0
samples = 3
value = [3, 0]
class = yes
gini = 0.0
samples = 6
value = [0, 6]
class = no

决策树

CART剪枝

1. 从生成算法产生的决策树T_0底端开始不断剪枝，直到T_0的根结点，形成子树序列$\{T_0, T_1, \ldots, T_n\}$
2. 通过交叉验证法在独立的验证数据集上对子树序列进行测试，从中选择最优子树

决策树

CART剪枝—形成一个子树序列

剪枝过程中，计算子树的损失函数

$$C_\alpha(T) = C(T) + \alpha|T|$$

其中，T为任意子树，C(T)为对训练数据的预测误差(如基尼指数)，|T|为子树的叶结点个数，$\alpha \geqslant 0$为参数，C_α(T)为参数是α时的子树T的整体损失。参数α权衡训练数据的拟合程度与模型的复杂度。

当α大的时候，最优子树T。偏小;当a小的时候，最优子树T_α偏大。极端情况，当α=0时，整体树是最优的。当$\alpha \to \infty$时，根结点组成的单结点树是最优的。

决策树

CART剪枝—形成一个子树序列

剪枝过程中，计算子树的损失函数

$$C_\alpha(T) = C(T) + \alpha|T|$$

从整体树T_0开始剪枝，对T_0的任意内部结点t,以t为单结点树的损失函数是

$$C_\alpha(t) = C(t) + \alpha$$

以t为根结点的子树T_t的损失函数是

$$C_\alpha(T_t) = C(T_t) + \alpha|T_t|$$

当α=0及α很小时 $C_\alpha(T_t) < C_\alpha(t)$

$$C(t) + \alpha = C(T_t) + \alpha|T_t|$$

$$C(t) - C(T_t) = \alpha(|T_t| - 1)$$

$$\alpha = \frac{C(t) - C(T_t)}{|T_t| - 1}$$

决策树

CART剪枝—形成一个子树序列

从整体树T_0开始剪枝，对T_0的任意内部结点t,以t为单结点树的损失函数是

$$C_\alpha(t) = C(t) + \alpha$$

以t为根结点的子树T_t的损失函数是

$$C_\alpha(T_t) = C(T_t) + \alpha|T_t|$$

$$\alpha = \frac{C(t) - C(T_t)}{|T_t| - 1}$$

对于当前这个结点，只要α≥该值，就一定有$C_\alpha(t) \le C_\alpha(T_t)$。
也就是剪掉这个结点比不剪更优

Breiman等人证明:可以用递归的方法对树进行剪枝。将α从小增大，$0 = \alpha_0 < \alpha_1 < \cdots < \alpha_n < +\infty$，
产生一系列的区间$[\alpha_i, \alpha_{i+1})$ $i = 0,1,2,\ldots,n$，剪枝得到的子树序列对应着区间$\alpha \in [\alpha_i, \alpha_{i+1})$，
$i = 0,1,2,\ldots,n$的最优子树序列$\{T_0, T_1, \ldots, T_n\}$

为此，对T_0中每一内部结点t,计算

$$g(t) = \frac{C(t) - C(T_t)}{|T_t| - 1}$$

它表示剪枝后整体损失函数减少的程度。在T_0中剪去g(t)最小的T_t,将得到的子树作为T,同时将最小的g(t)设为α_1。T1为区间$[\alpha_1, \alpha_2)$的最优子树。

决策树

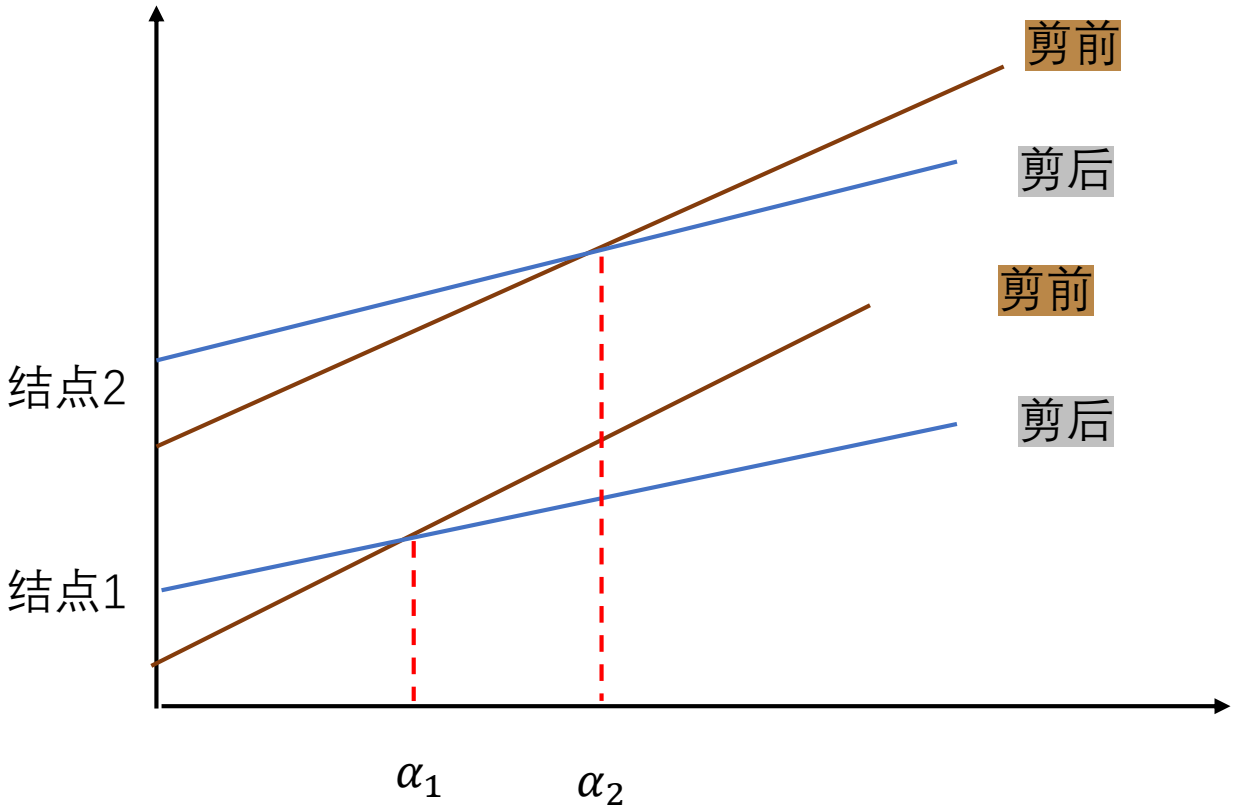
CART剪枝—形成一个子树序列

为此，对T_0中每一内部结点t,计算

$$g(t) = \frac{C(t) - C(T_t)}{|T_t| - 1}$$

因为我们现在改动的只是这个内部结点t,相当于把内部结点的子节点回退到它内部，形成一个子节点，而其他结点的损失函数并没有发生改变

它表示剪枝后整体损失函数减少的程度。在T_0中剪去g(t)最小的T_t,将得到的子树作为T,同时将最小的g(t)设为α_1。T1为区间$[\alpha_1, \alpha_2)$的最优子树。



以图中两个点为例，结点1和结点2，g(t)2大于g(t)1，假设在所有结点中g(t)1最小，g(t)2最大，两种选择方法：当选择最大值g(t)2，即结点2进行剪枝，但此时结点1的不修剪的误差大于修剪之后的误差，即如果不修剪的话，误差变大，依次类推，对其它所有的结点的g(t)都是如此，从而造成整体的累计误差更大。反之，如果选择最小值g(t)1，即结点1进行剪枝，则其余结点不剪的误差要小于剪后的误差，不修剪为好，且整体的误差最小。

决策树

CART剪枝---交叉验证

在剪枝得到的子树序列$T_0, T_1, \cdots, T_n$中通过交叉验证选取最优子树T_α

具体地，利用独立的验证数据集，测试子树序列$T_0, T_1, \cdots, T_n$中各棵子树的平方误差或基尼指数。平方误差或基尼指数最小的决策树被认为是最优的决策树。在子树序列中，每棵子树$T_0, T_1, \cdots, T_n$都对应于一个参数$\alpha_1, \alpha_2, \cdots, \alpha_n$。所以，当最优子树$T_k$确定时，对应的$\alpha_k$也确定了,即得到最优决策树$T_\alpha$。

算法 5.7（CART 剪枝算法）

输入：CART 算法生成的决策树 T_0；

输出：最优决策树 T_α。

（1）设 $k=0$，$T=T_0$。

（2）设 $\alpha=+\infty$。

（3）自下而上地对各内部结点 t 计算 $C(T_t)$，$|T_t|$ 以及

$$g(t)=\frac{C(t)-C(T_t)}{|T_t|-1}$$

$$\alpha=\min(\alpha,g(t))$$

这里，T_t 表示以 t 为根结点的子树，$C(T_t)$ 是对训练数据的预测误差，$|T_t|$ 是 T_t 的叶结点个数。

（4）对 $g(t)=\alpha$ 的内部结点 t 进行剪枝，并对叶结点 t 以多数表决法决定其类，得到树 T。

（5）设 $k=k+1$，$\alpha_k=\alpha$，$T_k=T$。

（6）如果 T_k 不是由根结点及两个叶结点构成的树，则回到步骤 (2)；否则令 $T_k=T_n$。

（7）采用交叉验证法在子树序列 $T_0,T_1,\cdots,T_n$ 中选取最优子树 T_α。 ■

THANKS