

# 第五章

# 决策树

张志忠 副教授

华东师范大学计算机学院

# 目录

1. 决策树模型与学习
2. 特征选择
3. 决策树的生成
4. 决策树的剪枝
5. CART算法

# 一、决策树的模型与学习

通过对分类问题一般方法的描述，可以看出分类问题一般包括两个步骤：

## 1、模型构建（归纳）

通过对训练集合的归纳，建立分类模型。

## 2、预测应用（推论）

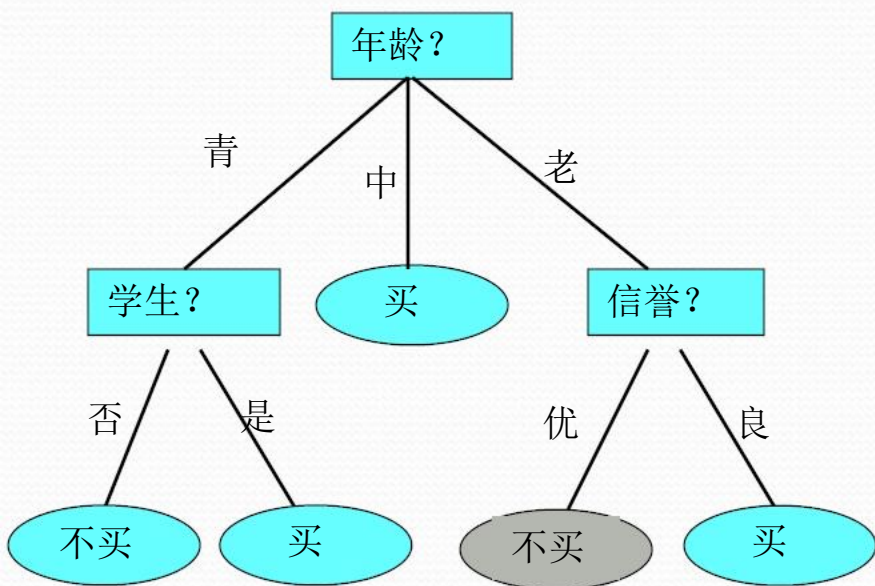
根据建立的分类模型，对测试集合进行测试。

# 决策树定义

- ❧ 分类决策树模型是一种描述对实例进行分类的树形结构
  - ❧ 结点（内部结点、叶结点）
  - ❧ 有向边或分支
- ❧ 本质上决策树是通过一系列规则对数据进行分类的过程。
  - ❧ If-then规则（由决策树的根结点到叶结点的每一条路径构建一条规则；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）
  - ❧ 互斥且完备：每一个实例都被一条路径或一条规则覆盖，而且只被一条路径或规则所覆盖

# 决策树的表示

决策树的基本组成部分：决策结点、分支和叶子。



决策树中最上面的结点称为根结点。是整个决策树的开始。每个分支是一个新的决策结点，或者是树的叶子。每个决策结点代表一个问题或者决策。通常对应待分类对象的属性。每个叶结点代表一种可能的分类结果

在沿着决策树从上到下的遍历过程中，在每个结点都有一个测试。对每个结点上问题的不同测试输出导致不同的分枝，最后会达到一个叶子结点。这一过程就是利用决策树进行分类的过程，利用若干个变量来判断属性的类别

# 决策树

- ❧ 决策树是一种典型的分类方法
  - ❧ 首先对数据进行处理，利用归纳算法生成可读的规则和决策树，
  - ❧ 然后使用决策对新数据进行分析。
- ❧ 本质上决策树是通过一系列规则对数据进行分类的过程。

# 决策树

## 决策树的优点

- 1、推理过程容易理解，决策推理过程可以表示成If Then形式；
- 2、推理过程完全依赖于属性变量的取值特点；
- 3、可自动忽略目标变量没有贡献的属性变量，也为判断属性变量的重要性，减少变量的数目提供参考。

# 决策树算法

与决策树相关的重要算法包括：

∞ ID<sub>3</sub>, C<sub>4.5</sub>, CART

∞ 算法的发展过程

∞ 1979年, J.R. Quinlan 给出ID<sub>3</sub>算法, 并在1983年和1986年对ID<sub>3</sub>进行了总结和简化, 使其成为决策树学习算法的典型。

∞ Schlimmer 和Fisher 于1986年对ID<sub>3</sub>进行改造, 在每个可能的决策树节点创建缓冲区, 使决策树可以递增式生成, 得到ID<sub>4</sub>算法。

∞ 1988年, Utgoff 在ID<sub>4</sub>基础上提出了ID<sub>5</sub>学习算法, 进一步提高了效率。

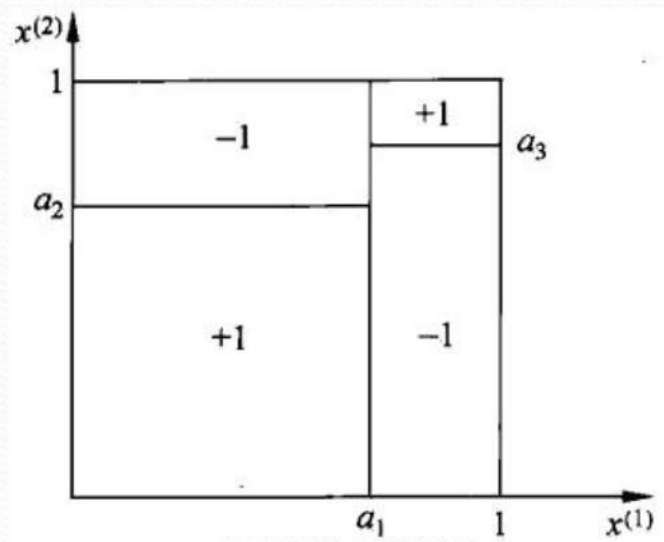
∞ 1993年, Quinlan 进一步发展了ID<sub>3</sub>算法, 改进成C<sub>4.5</sub>算法。

∞ 另一类决策树算法为CART, 与C<sub>4.5</sub>不同的是, CART的决策树由二元逻辑问题生成, 每个树节点只有两个分枝, 分别包括学习实例的正例与反例。

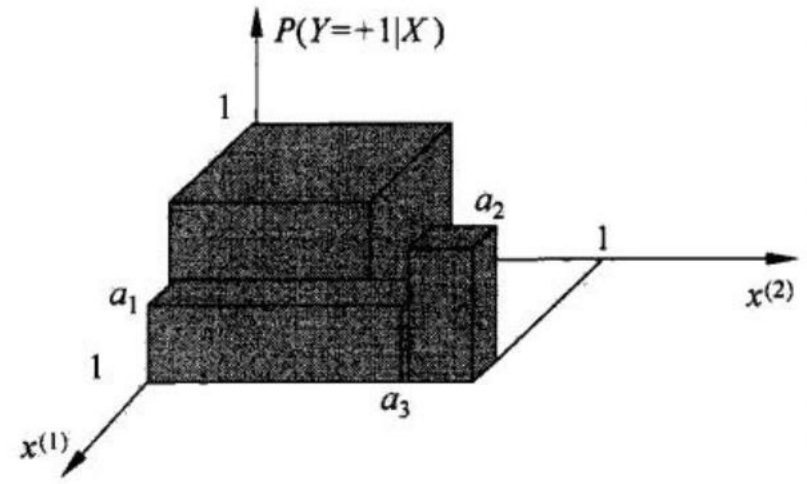
# 决策树与条件概率分布

- ❧ 决策树表示给定特征条件下类的条件概率分布。
- ❧ 条件概率分布定义在特征空间的一个划分(partition)上. 将特征空间划分为互不相交的单元(cell)或区域(region), 并在每个单元定义一个类的概率分布就构成了一个条件概率分布。
- ❧ 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

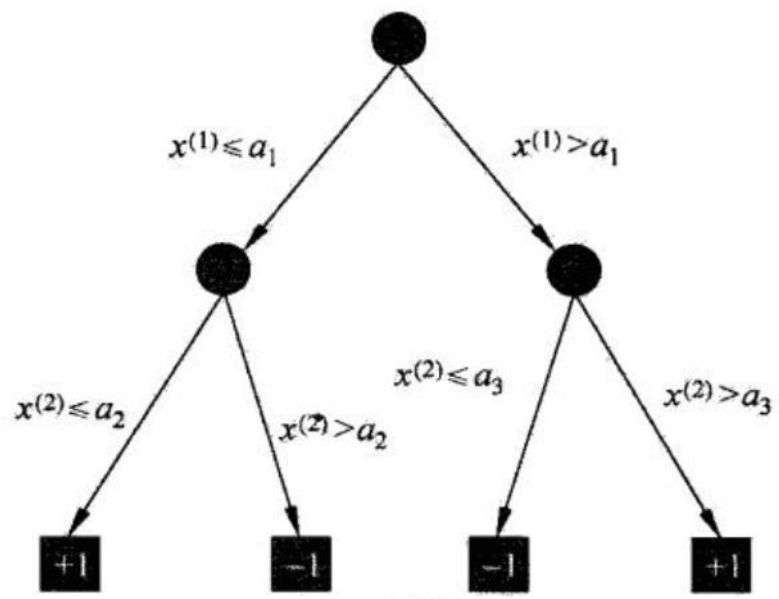
# 决策树与条件概率分布



(a) 特征空间划分



(b) 条件概率分布



(c) 决策树

# 决策树与条件概率分布

- ❧ 决策树学习本质上是从训练数据集中归纳出一组分类规则，与训练数据集不相矛盾的决策树。
- ❧ 能对训练数据进行正确分类的决策树可能有多个，也可能一个也没有.我们需要的是一个与训练数据矛盾较小的决策树，同时具有很好的泛化能力.
- ❧ 决策树学习是由训练数据集估计条件概率模型.基于特征空间划分的类的条件概率模型有无穷多个.
- ❧ 我们选择的条件概率模型应该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.

# 决策树与条件概率分布

如何开始？

**特征选择**：选择有代表性的特征，对空间进行划分

然后呢？

**决策树的生成**：局部选择

**决策树的剪枝**：全局最优

# 二、特征选择

∞ 信息增益

∞ 信息增益比

信息的本质是消除不确定性。

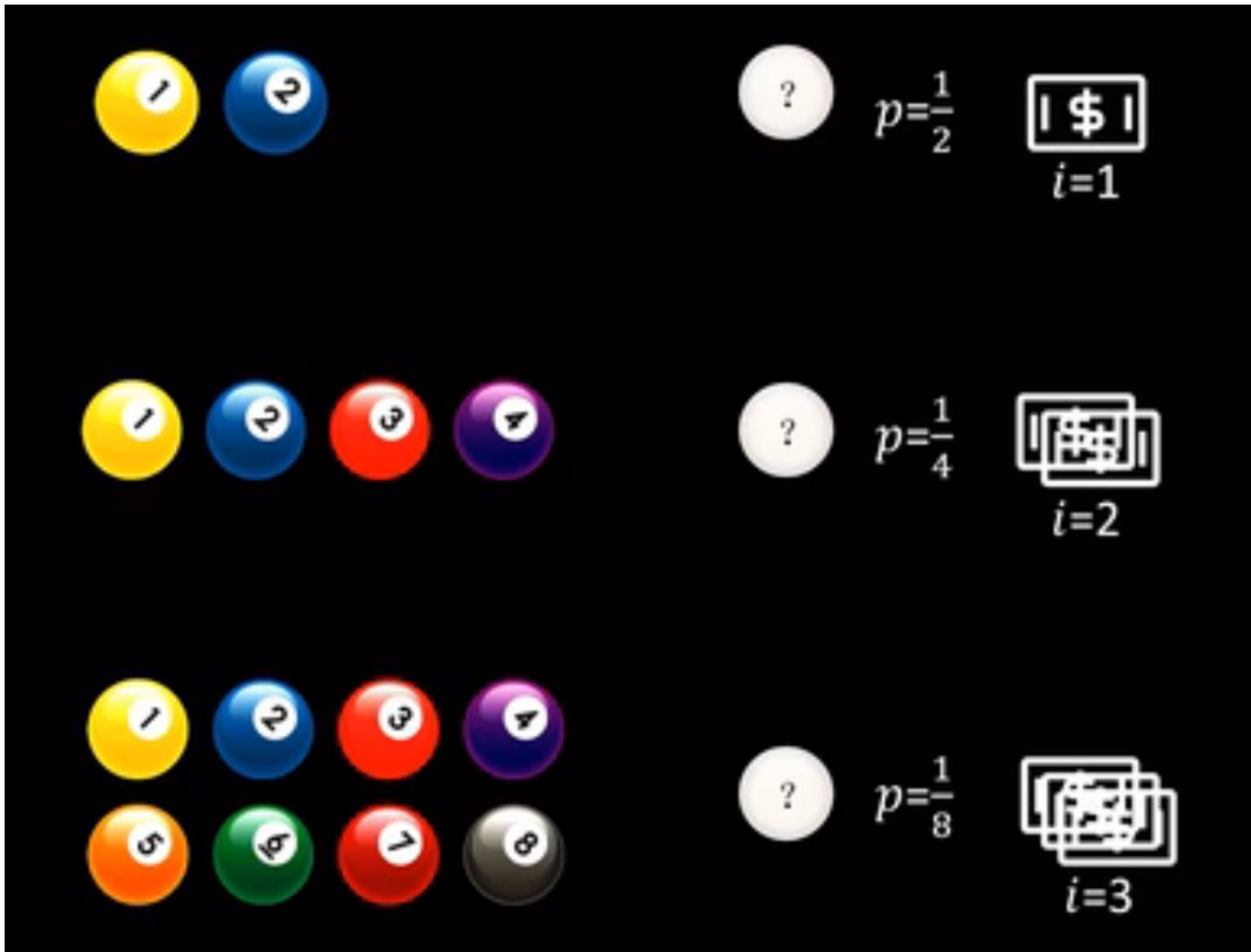


骰子的点数大于0

骰子的点数大于3

骰子的点数是5

信息的本质是消除不确定性。



$$I = -\log_2 p$$

信息的本质是消除不确定性。



骰子的点数大于0

$$p = 1$$
$$I = 0 \text{ bit}$$

骰子的点数大于3

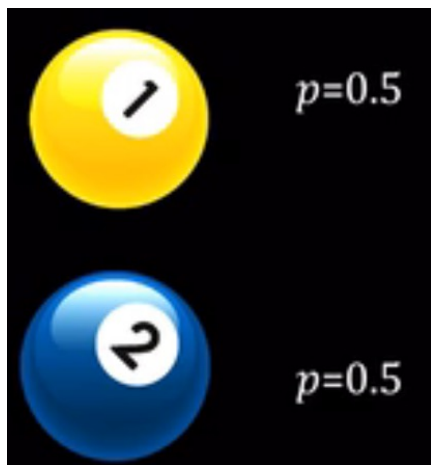
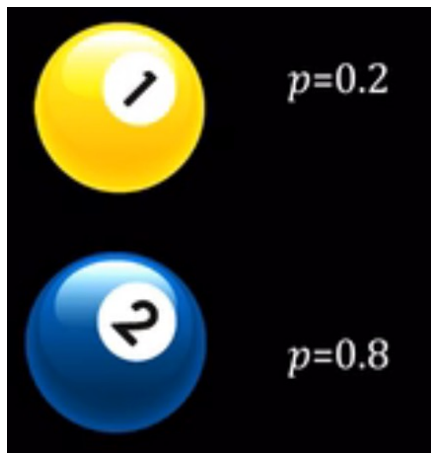
$$p = 1/2$$
$$I = 1 \text{ bit}$$

骰子的点数是5

$$p = 1/6$$
$$I = 2.58 \text{ bit}$$

$$I = -\log_2 p$$

信息的本质是消除不确定性。



# 信息增益

∞ 熵的理论解释:

∞ 设X是一个取有限个值的离散随机变量，其概率分布为:

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

∞ 则随机变量X的熵定义为:

$$H(X) = -\sum_{i=1}^n p_i \log p_i$$

∞ 对数以2为底或以e为底(自然对数), 这时熵的单位分别称作比特(bit)或纳特(nat), 熵只依赖于X的分布, 与X的取值无关。

$$H(p) = -\sum_{i=1}^n p_i \log p_i$$

# 信息增益

∞ 熵的理论解释:

∞ 熵越大, 随机变量的不确定性越大:

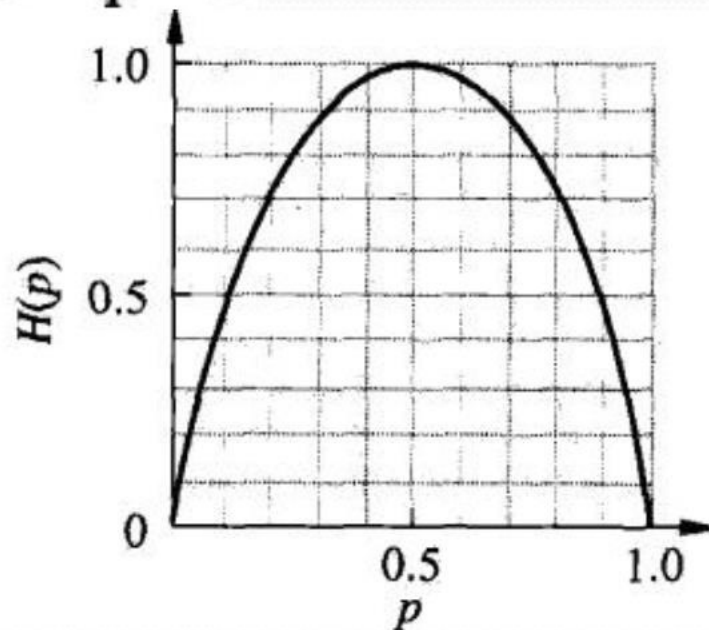
$$0 \leq H(p) \leq \log n$$

∞ 当X为1,0分布时:

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

∞ 熵:

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



# 信息增益

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, 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:

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

当熵和条件熵中的概率由数据估计 (特别是极大似然估计) 得到时, 所对应的熵与条件熵分别称为经验熵 (empirical entropy) 和经验条件熵 (empirical conditional entropy)

# 信息增益

定义5.2 (信息增益):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

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

(Information gain)表示得知特征X的信息而使得类Y的信息的不确定性减少的程度.

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.

# 信息增益的算法

- 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 $C_k$ ,  $k = 1, 2, \dots, K$ ,
- $|C_k|$ 为属于类 $C_k$ 的样本个数
- 特征 $A$ 有 $n$ 个不同的取值 $\{a_1, a_2, \dots, a_n\}$ 根据特征 $A$ 的取值将 $D$ 划分为 $n$ 个子集 $D_1, \dots, D_n$
- $|D_i|$ 为 $D_i$ 的样本个数
- 记子集 $D_i$ 中属于类 $C_k$ 的样本集合为 $D_{ik}$
- $|D_{ik}|$ 为 $D_{ik}$ 的样本个数

# 信息增益的算法

☞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

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

# 三、决策树的生成

☞ ID<sub>3</sub>算法

☞ ID<sub>3</sub>算法-流程

☞ ID<sub>3</sub>算法-实际使用

☞ ID<sub>3</sub>算法-小结

☞ C<sub>4.5</sub>的生成算法

# 决策树ID3算法

- ❧ ID<sub>3</sub>算法是一种经典的决策树学习算法，由Quinlan于1979年提出。
- ❧ ID<sub>3</sub>算法主要针对属性选择问题。是决策树学习方法中最具影响和最为典型的算法。
- ❧ 该方法使用信息增益度选择测试属性。
- ❧ 当获取信息时，将不确定的内容转为确定的内容，因此信息伴着不确定性。
- ❧ 从直觉上讲，小概率事件比大概率事件包含的信息量大。如果某件事情是“百年一见”则肯定比“习以为常”的事件包含的信息量大。
- ❧ 如何度量信息量的大小？  
答：信息增益

# 决策树ID3算法

✧ 在决策树分类中，假设 $S$ 是训练样本集合， $|S|$ 是训练样本数，样本划分为 $n$ 个不同的类 $C_1, C_2, \dots, C_n$ ，这些类的大小分别标记为 $|C_1|, |C_2|, \dots, |C_n|$ 。则任意样本 $S$ 属于类 $C_i$ 的概率为：

$$p(S_i) = \frac{|C_i|}{|S|}$$

$$\text{Entropy}(S|A) = \sum (|S_v|/|S|) * \text{Entropy}(S_v)$$

✧  $\Sigma$ 是属性 $A$ 的所有可能的值 $v$ ， $S_v$ 是属性 $A$ 有 $v$ 值的 $S$ 子集

✧  $|S_v|$ 是 $S_v$ 中元素的个数； $|S|$ 是 $S$ 中元素的个数。

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

## 第1步计算决策属性的熵

决策属性“买计算机？”。  
该属性分两类：买/不买

$$|C_1|(\text{买})=641$$

$$|C_2|(\text{不买})=383$$

$$|D|=|C_1|+|C_2|=1024$$

$$P_1=641/1024=0.6260$$

$$P_2=383/1024=0.3740$$

$$\begin{aligned} H(D) &= -P_1 \log_2 P_1 - P_2 \log_2 P_2 \\ &= -(P_1 \log_2 P_1 + P_2 \log_2 P_2) \\ &= 0.9537 \end{aligned}$$

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

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

## 第2步计算条件属性的熵

条件属性共有4个：  
年龄、收入、学生、信誉。  
分别计算不同属性的信息增益。

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

## 第2-1步计算年龄的熵

年龄共分三个组：

青年、中年、老年

青年买与不买比例为128/256

$$|D_{11}|(\text{买})=128$$

$$|D_{12}|(\text{不买})=256$$

$$|D_1|=384$$

$$P_1=128/384$$

$$P_2=256/384$$

$$H(D_1)=-P_1\log_2 P_1 - P_2\log_2 P_2$$

$$=-(P_1\log_2 P_1 + P_2\log_2 P_2)$$

$$=0.9183$$

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

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

## 第2-2步计算年龄的熵

年龄共分三个组：

青年、中年、老年

中年买与不买比例为256/0

$$|D_{21}|(\text{买})=256$$

$$|D_{22}|(\text{不买})=0$$

$$|D_2|=256$$

$$P_1=256/256$$

$$P_2=0/256$$

$$\begin{aligned} H(D_2) &= -P_1 \log_2 P_1 - P_2 \log_2 P_2 \\ &= -(P_1 \log_2 P_1 + P_2 \log_2 P_2) \\ &= 0 \end{aligned}$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

## 第2-3步计算年龄的熵

年龄共分三个组：

青年、中年、老年

老年买与不买比例为125/127

$$|D_{31}|(\text{买})=125$$

$$|D_{32}|(\text{不买})=127$$

$$|D_3|=S_1+S_2=252$$

$$P_1=125/252$$

$$P_2=127/252$$

$$H(D_3)=-P_1\log_2 P_1 - P_2\log_2 P_2$$

$$=-(P_1\log_2 P_1 + P_2\log_2 P_2)$$

$$=0.9157$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

## 第2-4步计算年龄的熵

年龄共分三个组：

青年、中年、老年

所占比例

青年组  $384/1024=0.375$

中年组  $256/1024=0.25$

老年组  $384/1024=0.375$

计算年龄的平均信息期望

$$\begin{aligned}
 E(\text{年龄}) &= 0.375 * 0.9183 + \\
 &\quad 0.25 * 0 + \\
 &\quad 0.375 * 0.9157 \\
 &= 0.6877
 \end{aligned}$$

G (年龄信息增益)

$$\begin{aligned}
 &= 0.9537 - 0.6877 \\
 &= 0.2660 \quad (1)
 \end{aligned}$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

### 第3步 计算收入的熵

收入共分三个组：  
高、中、低

$$E(\text{收入}) = 0.9361$$

$$\begin{aligned} \text{收入信息增益} &= 0.9537 - 0.9361 \\ &= 0.0176 \quad (2) \end{aligned}$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

## 第4步计算学生的熵

学生共分二个组：

学生、非学生

$$E(\text{学生}) = 0.7811$$

$$\begin{aligned} \text{年龄信息增益} &= 0.9537 - 0.7811 \\ &= 0.1726 \quad (3) \end{aligned}$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

## 第5步计算信誉的熵

信誉分二个组：

良好，优秀

$$E(\text{信誉}) = 0.9048$$

$$\begin{aligned} \text{信誉信息增益} &= 0.9537 - 0.9048 \\ &= 0.0453 \quad (4) \end{aligned}$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

## 第6步计算选择节点

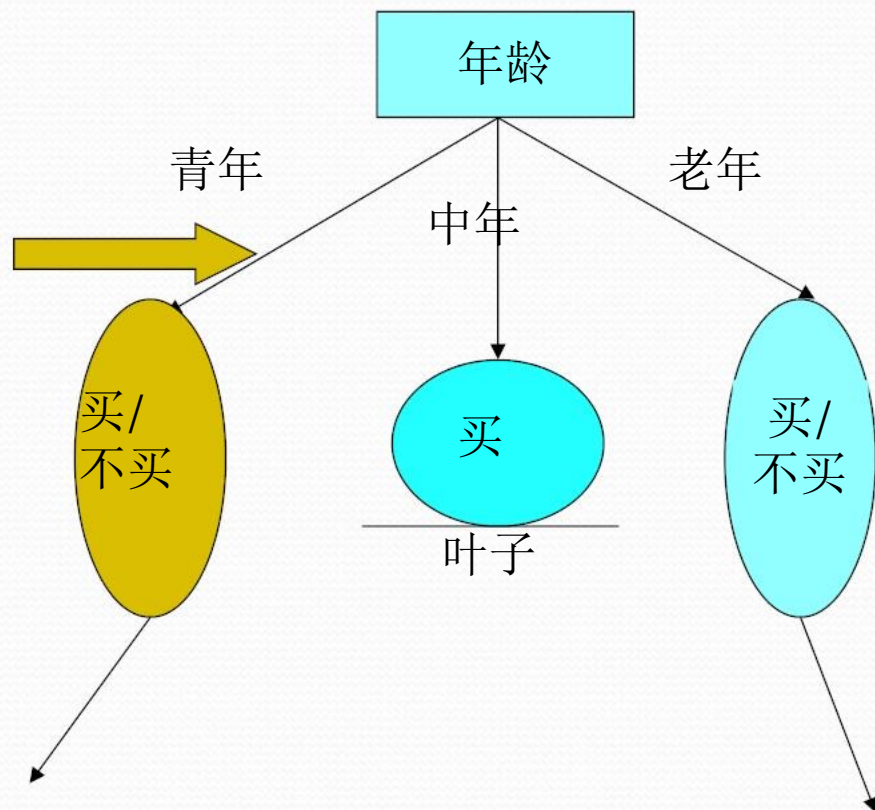
$$\begin{aligned} \text{年龄信息增益} &= 0.9537 - 0.6877 \\ &= 0.2660 \quad (1) \end{aligned}$$

$$\begin{aligned} \text{收入信息增益} &= 0.9537 - 0.9361 \\ &= 0.0176 \quad (2) \end{aligned}$$

$$\begin{aligned} \text{年龄信息增益} &= 0.9537 - 0.7811 \\ &= 0.1726 \quad (3) \end{aligned}$$

$$\begin{aligned} \text{信誉信息增益} &= 0.9537 - 0.9048 \\ &= 0.0453 \quad (4) \end{aligned}$$

计数	年龄	收入	学生	信誉	归类: 买计算机?
64	青	高	否	良	不买
64	青	高	否	优	不买
12 8	青	中	否	良	不买
64	青	低	是	良	买
64	青	中	是	优	买



计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	青	中	否	良	不买
64	青	低	是	良	买
64	青	中	是	优	买

青年买与不买比例为  
128/256

$$|C_1|(\text{买})=128$$

$$|C_2|(\text{不买})=256$$

$$|D|=384$$

$$P_1=128/384$$

$$P_2=256/384$$

$$\begin{aligned} H(D) &= -P_1 \log_2 P_1 - P_2 \log_2 P_2 \\ &= -(P_1 \log_2 P_1 + P_2 \log_2 P_2) \\ &= 0.9183 \end{aligned}$$

计数	年龄	收入	学生	信誉	归类: 买计算机?
64	青	高	否	良	不买
64	青	高	否	优	不买
128	青	中	否	良	不买
64	青	低	是	良	买
64	青	中	是	优	买

如果选择收入作为节点  
分高、中、低

$$H(D1)=0$$

比例:  
 $128/384=0.3333$

$$H(D2)=0.9183$$

比例:  $192/384=0.5$

$$H(D3)=0$$

比例:  $64/384=0.1667$

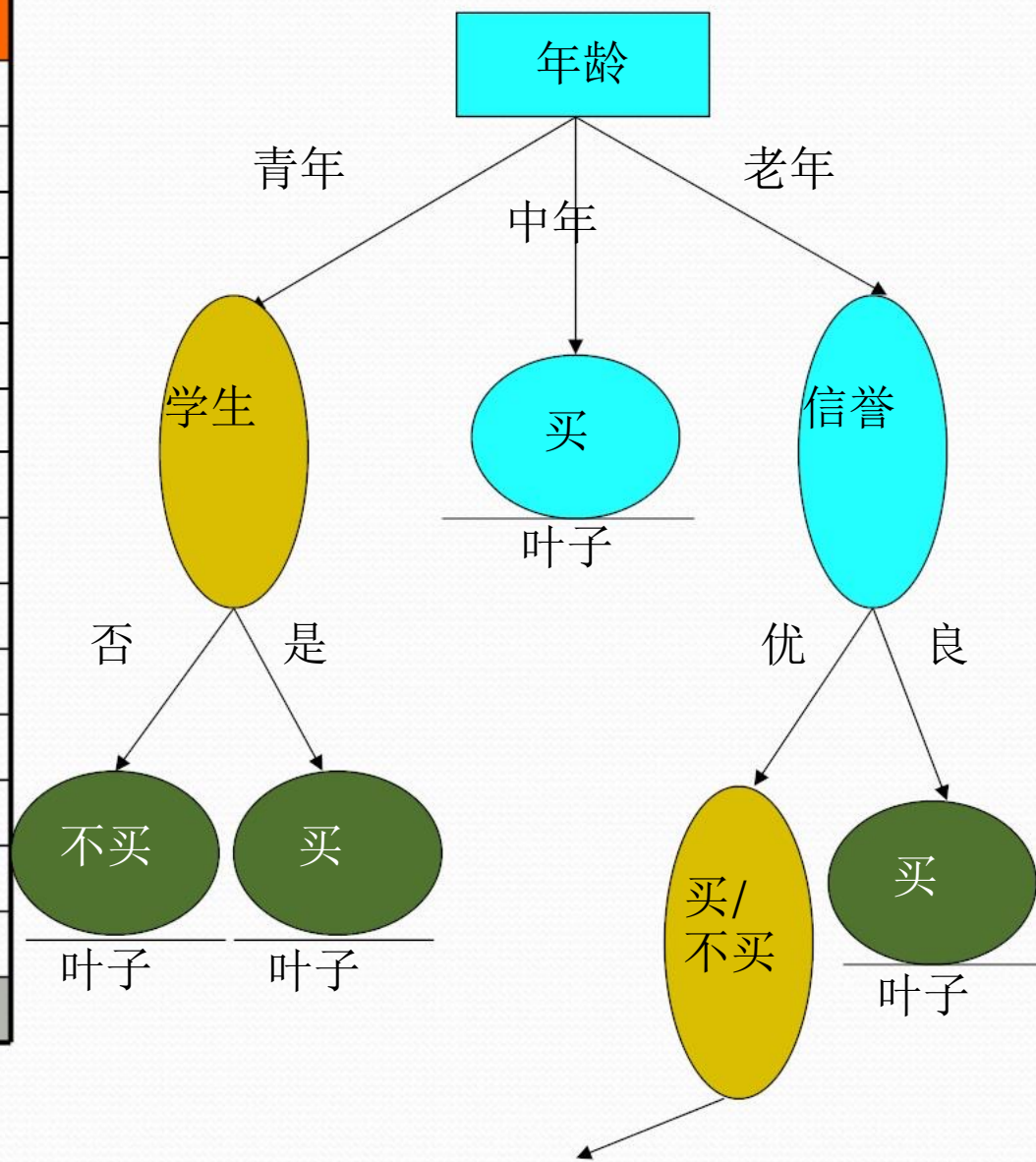
平均信息期望 (加权总和) :

$$E(\text{收入}) = 0.3333 * 0 + 0.5 * 0.9183 + 0.1667 * 0 = 0.4592$$

$$\text{Gain}(\text{收入}) = I(128, 256) - E(\text{收入}) = 0.9183 - 0.4592 = 0.4591$$

注意

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买



# 决策树ID3算法-流程

- ❧<sub>1</sub> 决定分类属性；
- ❧<sub>2</sub> 对目前的数据表，建立一个节点N
- ❧<sub>3</sub> 如果数据库中的数据都属于同一个类，N就是树叶，在树叶上标出所属的类
- ❧<sub>4</sub> 如果数据表中没有其他属性可以考虑，则N也是树叶，按照少数服从多数的原则在树叶上标出所属类别
- ❧<sub>5</sub> 否则，根据平均信息期望值E或GAIN值选出一个最佳属性作为节点N的测试属性
- ❧<sub>6</sub> 节点属性选定后，对于该属性中的每个值：
  - ❧ 从N生成一个分支，并将数据表中与该分支有关的数据收集成分支节点的数据表，在表中删除节点属性那一栏如果分支数据表非空，则运用以上算法从该节点建立子树。



# 决策树ID3算法-实际使用

## 决策树的数据准备

### 整理后的数据表

计数	年龄	收入	学生	信誉	归类： 买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
。。					
。					

- ❖ **Data cleaning**  
删除/减少noise,  
补填missing values
- ❖ **Data transformation**  
数据标准化 (data normalization)  
数据归纳 (generalize data to higher-level concepts using concept hierarchies)  
例如：年龄归纳为老、中、青三类  
控制每个属性的可能值不超过七种  
(最好不超过五种)
- ❖ **Relevance analysis**  
对于与问题无关的属性：删  
对于属性的可能值大于七种  
又不能归纳的属性：删

# 决策树ID3算法-小结

✎ ID<sub>3</sub>算法的基本思想是，以信息熵为度量，用于决策树节点的属性选择，每次优先选取信息量最多的属性，亦即使熵值变为最小的属性，以构造一颗熵值下降最快的决策树，到叶子节点处的熵值为0。此时，每个叶子节点对应的实例集中的实例属于同一类。

# 决策树C4.5的生成算法

C4.5 算法与 ID3 算法相似，C4.5 算法对 ID3 算法进行了改进。C4.5 在生成的过程中，用信息增益比来选择特征。

$$g_R(D, A) = \frac{g(D, A)}{H_A(D)}$$

## 算法 5.3 (C4.5 的生成算法)

输入：训练数据集  $D$ ，特征集  $A$ ，阈值  $\epsilon$ ；

输出：决策树  $T$ 。

$$H_A(D) = - \sum_{i=1}^n \frac{|D_i|}{|D|} \log_2 \frac{|D_i|}{|D|}, \quad n \text{ 是特征 } A \text{ 取值的个数}$$

(1) 如果  $D$  中所有实例属于同一类  $C_k$ ，则置  $T$  为单结点树，并将  $C_k$  作为该结点的类，返回  $T$ ；

(2) 如果  $A = \emptyset$ ，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，返回  $T$ ；

(3) 否则，按式(5.10)计算  $A$  中各特征对  $D$  的信息增益比，选择信息增益比最大的特征  $A_g$ ；

(4) 如果  $A_g$  的信息增益比小于阈值  $\epsilon$ ，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，返回  $T$ ；

(5) 否则，对  $A_g$  的每一可能值  $a_i$ ，依  $A_g = a_i$  将  $D$  分割为子集若干非空  $D_i$ ，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) 对结点  $i$ ，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  $T_i$ 。



# 四、决策树的剪枝

决策树的剪枝

# 决策树的剪枝

- 通过极小化决策树整体的损失函数或代价函数来实现。
- 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,  $\alpha \geq 0$ 为参数, 损失函数:

经验熵:

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

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

原式第一项:

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

则:

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

# 决策树的剪枝

树的剪枝算法：

输入：生成算法产生的整个树  $T$ ，参数  $\alpha$ ；

输出：修剪后的子树  $T_\alpha$ 。

(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和的损失函数分别为： $C_\alpha(T_B)$ 与 $C_\alpha(T_A)$ 。

如果： $C_\alpha(T_A) \leq C_\alpha(T_B)$  剪枝

(3) 返回 (2)，直至不能继续为止，得到损失函数最小的子树

# 五、CART算法

☞ 决策树面临的问题

☞ CART生成

☞ CART剪枝

# 决策树面临的问题

理想的决策树有三种：

⌘ (1)叶子结点数最少；

⌘ (2)叶子结点深度最小；

⌘ (3)叶子结点数最少且叶子结点深度最小。

⌘ 然而，洪家荣等人已经证明了要找到这种最优的决策树是NP难题。因此，决策树优化的目的就是要找到尽可能趋向于最优的决策树。

# 决策树面临的问题

## ❧ 过度拟合

❧ 决策树算法增长树的每一个分支的深度，直到恰好能对训练样例比较完美地分类。实际应用中，当数据中有噪声或训练样例的数量太少以至于不能产生目标函数的有代表性的采样时，该策略可能会遇到困难。

❧ 在以上情况发生时，这个简单的算法产生的树会过拟合训练样例（过度拟合：Over Fitting）。

# 决策树面临的问题

## ❧ 过度拟合

- ❧ 对学习算法是否成功的真正测试是看它对于训练中未见到的数据的执行性能。训练过程应该包含训练样本和验证样本。验证样本用于测试训练后的性能。如果验证结果差，则需要考虑采用不同的结构重新进行训练，例如使用更大的样本集，或者改变从连续值到离散值得数据转换等。
- ❧ 通常应该建立一个验证过程，在训练最终完成后用来检测训练结果的泛化能力。

# 决策树面临的问题

✎ 一般可以将分类模型的误差分为：

✎1、训练误差（Training Error）；

✎2、泛化误差（Generalization Error）

✎ 训练误差是在训练记录上误分类样本比例；

✎ 泛化误差是模型在未知记录上的期望误差；

✎ 一个好的模型不仅要能够很好地拟合训练数据，而且对未知样本也要能够准确地分类。

✎ 一个好的分类模型必须具有低的训练误差和泛化误差。因为一个具有低训练误差的模型，其泛化误差可能比具有较高训练误差的模型高。（训练误差低，泛化误差高，称为过渡拟合）

# 决策树面临的问题

- ❧ 决策树算法比较适合处理离散数值的属性。实际应用中属性是连续的或者离散的情况都比较常见。
- ❧ 在应用连续属性值时，在一个树结点可以将属性 $A_i$ 的值划分为几个区间。然后信息增益的计算就可以采用和离散值处理一样的方法。原则上可以将 $A_i$ 的属性划分为任意数目的空间。C4.5中采用的是二元分割（Binary Split）。需要找出一个合适的分割阈值。
- ❧ 本书第九章将介绍  
参考C4.5算法 Top 10 algorithms in data mining  
Knowledge Information System 2008 14:1-37

# CART树

- ∞ 目标变量是类别的 --- 分类树
- ∞ 目标变量是连续的 --- 回归树

# CART与ID3的不同

## ⌘二元划分：

⌘二叉树不易产生数据碎片，精确度往往也会高于多叉树

## ⌘CART中选择变量的不纯度度量：

⌘分类目标：Gini指标、Toving、order Toving

⌘连续目标：最小平方残差、最小绝对残差

## ⌘剪枝：

⌘用**预剪枝或后剪枝**对训练集生长的树进行剪枝

## ⌘树的建立：

⌘如果目标变量是标称的，并且是具有两个以上的类别，则CART可能考虑将目标类别合并成两个超类别（双化）；

⌘如果目标变量是连续的，则CART算法找出一组基于树的回归方程来预测目标变量。

# CART树

- ✧ CART算法由两部分组成：
  - ✧ 决策树生成
  - ✧ 决策树剪枝
- ✧ 回归树：平方误差最小化
- ✧ 分类树：Gini Index

# CART生成

∞ 回归树的生成

∞ 设Y是连续变量，给定训练数据集：

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

∞ 假设已将输入空间划分为M各单元 $R_1, R_2, R_m$ , 并且每个单元 $R_m$ 上有一个固定的输出 $C_m$ , 回归树表示为：

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

∞ 平方误差来表示预测误差，用平方误差最小准则求解每个单元上的最优输出值

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

∞  $R_m$ 上的 $C_m$ 的最优值： $\hat{c}_m = \text{ave}(y_i | x_i \in R_m)$

# CART生成

问题：如何对输入空间进行划分？

启发式：选择第 $j$ 个变量 $x^{(j)}$ 和它取的值 $s$ ，作为切分变量和切分点，定义两个区域：

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

然后寻找最优切分变量和切分点：

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

且： $\hat{c}_1 = \text{ave}(y_i | x_i \in R_1(j, s))$  和  $\hat{c}_2 = \text{ave}(y_i | x_i \in R_2(j, s))$

再对两个区域重复上述划分，直到满足停止条件。

# CART生成

## 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

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

遍历变量  $j$ ，对固定的切分变量  $j$  扫描切分点  $s$ ，选择使式 (1) 达到最小值的对  $(j,s)$

# CART生成

## 最小二乘回归树生成算法

(2) 用选定的对  $(j, s)$  划分区域并决定相应的输出值:

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

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

(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  $R_1, R_2, \dots, R_M$ , 生成决策树:

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

# CART生成

∞ 分类树的生成:

∞ 基尼指数

∞ 分类问题中，假设有k个类，样本点属于k的概率 $P_k$ ，则概率分布的基尼指数：

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

∞ 二分类问题： $\text{Gini}(p) = 2p(1-p)$

∞ 对给定的样本集合D，基尼指数

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

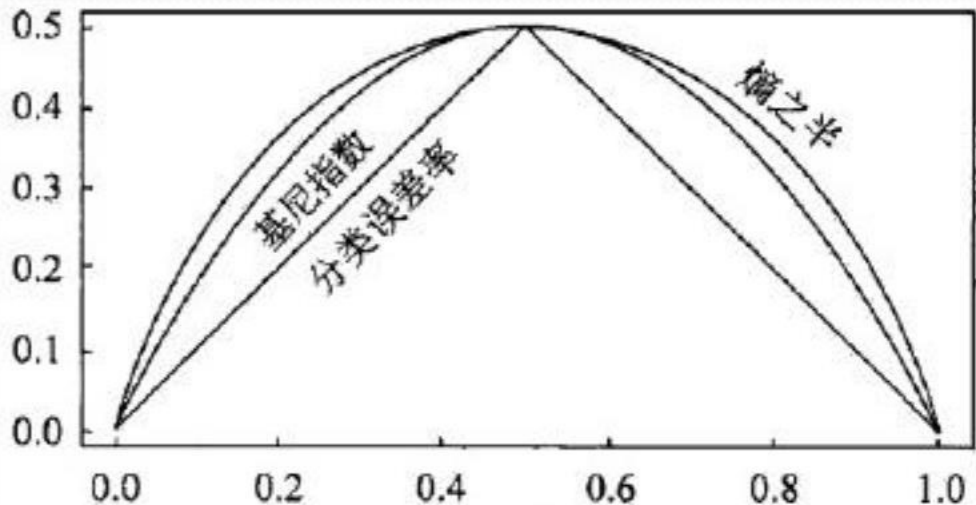
# CART生成

如果样本集合D根据特征A是否为a被分割成D<sub>1</sub>和D<sub>2</sub>，  
即

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

则在特征A的条件下，集合D的基尼指数：

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



# CART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 $D_1$ ， $D_2$ ，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

☞ 4、生成CART树

# CART剪枝

☞ CART剪枝

☞ 两步

- ☞ 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 $\{T_0, T_1..T_n\}$ ,
- ☞ 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

# CART剪枝

∞ CART剪枝

∞ 1、剪枝，形成子树序列

∞ 剪枝过程中，计算子树的损失函数：

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

∞ 对固定的 $\alpha$ 一定存在损失函数最小的子树，表示为 $T_{\alpha}$ ，

∞ 当 $\alpha$ 变大时，最优子树 $T_{\alpha}$ 偏小，

∞  $\alpha=0$ 时，整体树最优， $\alpha$ 趋近无穷大，单结点最优

∞ 将 $\alpha$ 从小增大， $0 = \alpha_0 < \alpha_1 < \dots < \alpha_n < +\infty$

最优子树序列  $\{T_0, T_1, \dots, T_n\}$

# CART剪枝

☞ CART剪枝

☞ 1、剪枝，形成子树序列

☞ 具体：从 $T_0$ 开始剪枝，以 $t$ 为单结点树的损失函数：

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

☞ 以 $t$ 为根结点的子树 $T_t$ 的损失函数：

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

☞ 当 $\alpha=0$ 及 $\alpha$ 很小时，

$$C_\alpha(T_t) < C_\alpha(t)$$

☞ 不断增大 $\alpha$ ，当

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

☞  $T_t$ 与 $t$ 有相同损失函数值，但 $t$ 结点更少，所以剪枝 $T_t$ 。

# CART剪枝

∞ CART剪枝

∞ 1、剪枝，形成子树序列

∞ 对 $T_0$ 中每个内部结点 $t$ ，计算：

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

∞ 在 $T_0$ 中剪去 $g(t)$ 最小的 $T_t$ ，将得到的子树作为 $T_1$ ，同时将最小的 $g(t)$ 设为 $a_1$ ， $T_1$ 为区间 $[a_1, a_2)$ 的最优子树

∞ 如此剪枝下去，直到根节点，不断增加 $a$ 的值，产生新的区间。

# CART剪枝

## ☞ CART剪枝

☞ 2、在剪枝得到的子树序列 $\{T_0, T_1 \dots T_n\}$ 中通过交叉验证选取最优子树 $T_a$

☞ 利用独立的验证数据集，测试子树序列中各子树的平方误差或基尼指数，最小的决策树就是最优决策树。



Q&A