

algorithm

## 4.分治策略e

决策树-ID3、C4.5、CART

# 决策树ID3算法

- ID3算法是一种经典的决策树学习算法，由Quinlan于1979年提出。
- ID3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？

# 信息增益

- Shannon 1948年提出的信息论理论:
- 熵(entropy): 信息量大小的度量, 即表示随机变量不确定性的度量。
- 熵的**通俗解释**: 事件 $a_i$ 的信息量 $I(a_i)$ 可如下度量:

$$I(a_i) = p(a_i) \log_2 \frac{1}{p(a_i)}$$

- 其中 $p(a_i)$ 表示事件 $a_i$ 发生的概率。
- 假设有 $n$ 个互不相容的事件 $a_1, a_2, a_3, \dots, a_n$ , 它们中有且仅有一个发生, 则其平均的信息量(熵)可如下度量:

$$I(a_1, a_2, \dots, a_n) = \sum_{i=1}^n I(a_i) = \sum_{i=1}^n p(a_i) \log_2 \frac{1}{p(a_i)}$$

# 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:

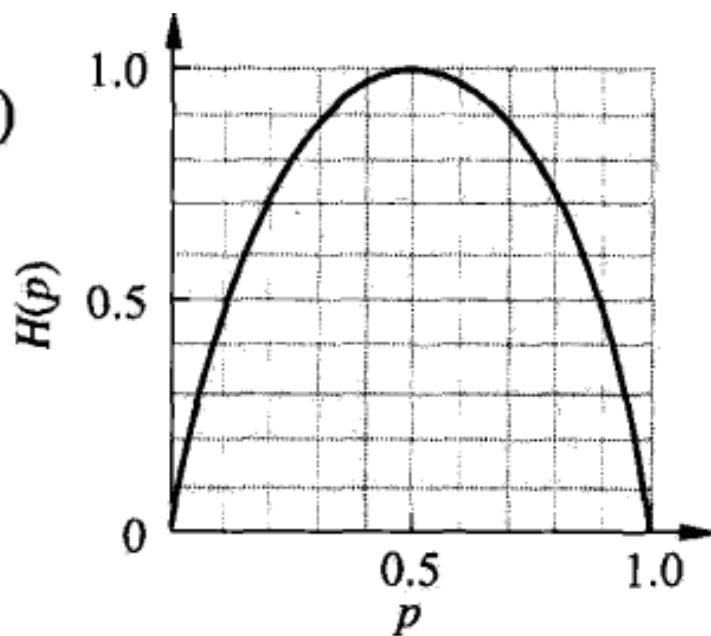
$$0 \leq H(p) \leq \log n$$

- 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

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

# 信息增益比

- 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
- 使用信息增益比可以对这一问题进行校正
- 定义5.3（信息增益比） 特征A对训练数据集D的信息增益比定义为信息增益与训练数据集D关于特征A的值的熵之比

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

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

# 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C1,C2,...Cn，这些类的大小分别标记为|C1|, |C2|, .....,|Cn|。则任意样本S属于类Ci的概率为：

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

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

- Σ是属性A的所有可能的值v
- Sv是属性A有v值的S子集
- |Sv|是Sv 中元素的个数
- |S|是S中元素的个数

# 决策树ID3算法计算步骤

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算法计算步骤

- 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

$$|C1|(\text{买})=641$$

$$|C2|(\text{不买})=383$$

$$|D|=|C1|+|C2|=1024$$

$$P1=641/1024=0.6260$$

$$P2=383/1024=0.3740$$

$$\begin{aligned} H(D) &= -P1\log_2 P1 - P2\log_2 P2 \\ &= -(P1\log_2 P1 + P2\log_2 P2) \\ &= 0.9537 \end{aligned}$$

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

# 决策树ID3算法计算步骤

- 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

# 决策树ID3算法计算步骤

- 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

$$|D_{11}|(\text{买})=128$$

$$|D_{12}|(\text{不买})=256$$

$$|D_1|=384$$

$$P_1=128/384$$

$$P_2=256/384$$

$$\begin{aligned} H(D_1) &= -P_1 \log_2 P_1 - P_2 \log_2 P_2 \\ &= -(P_1 \log_2 P_1 + P_2 \log_2 P_2) \\ &= 0.9183 \end{aligned}$$

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

# 决策树ID3算法计算步骤

- 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



# 决策树ID3算法计算步骤

- 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

$$|D_{31}|(\text{买})=125$$

$$|D_{32}|(\text{不买})=127$$

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

$$P_1=125/252$$

$$P_2=127/252$$

$$\begin{aligned} H(D_3) &= -P_1 \log_2 P_1 - P_2 \log_2 P_2 \\ &= -(P_1 \log_2 P_1 + P_2 \log_2 P_2) \\ &= 0.9157 \end{aligned}$$

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

# 决策树ID3算法计算步骤

- 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  $384/1025=0.375$
  - 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
$$E(\text{年龄}) = 0.375 \times 0.9183 + 0.25 \times 0 + 0.375 \times 0.9157 = 0.6877$$
$$G(\text{年龄信息增益}) = 0.9537 - 0.6877 = 0.2660 \quad (1)$$

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算法计算步骤

- 第3步 计算收入的熵
- 收入共分三个组：
  - 高、中、低
- $E(\text{收入}) = 0.9361$
- 收入信息增益  $= 0.9537 - 0.9361 = 0.0176$  (2)

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算法计算步骤

- 第4步计算学生的熵
- 学生共分二个组：
  - 学生、非学生
- $E(\text{学生}) = 0.7811$
- 学生信息增益  $= 0.9537 - 0.7811 = 0.1726$  (3)

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算法计算步骤

- 第5步计算信誉的熵
- 信誉分二个组：
  - 良好，优秀
- $E(\text{信誉}) = 0.9048$
- 信誉信息增益 =  $0.9537 - 0.9048 = 0.0453$  (4)

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

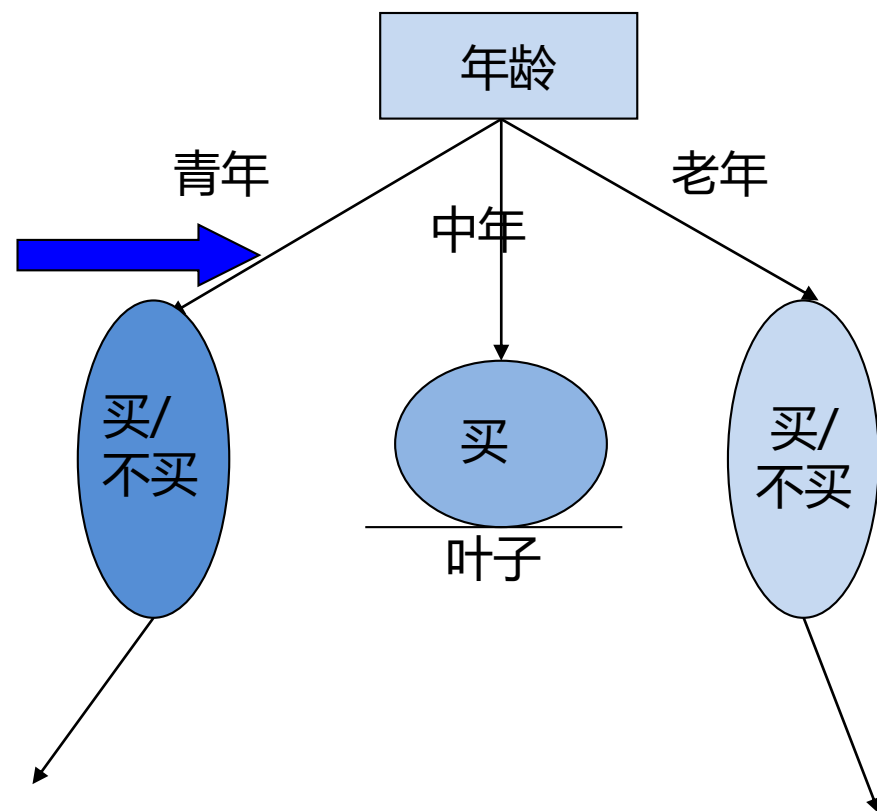
# 决策树ID3算法计算步骤

- 第6步计算选择节点
- 年龄信息增益 =  $0.9537 - 0.6877$   
= 0.2660 (1)
- 收入信息增益 =  $0.9537 - 0.9361$   
= 0.0176 (2)
- 年龄信息增益 =  $0.9537 - 0.7811$   
= 0.1726 (3)
- 信誉信息增益 =  $0.9537 - 0.9048$   
= 0.0453 (4)

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算法计算步骤

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



# 决策树ID3算法计算步骤

- 青年买与不买比例为128/256

$$|C1|(\text{买})=128$$

$$|C2|(\text{不买})=256$$

$$|D|=384$$

$$P1=128/384$$

$$P2=256/384$$

$$\begin{aligned} H(D) &= -P1\log_2 P1 - P2\log_2 P2 \\ &= -(P1\log_2 P1 + P2\log_2 P2) \\ &= 0.9183 \end{aligned}$$

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



# 决策树ID3算法计算步骤

- 如果选择收入作为节点
- 分高、中、低

$$H(D1)=0$$

$$\text{比例: } 128/384=0.3333$$

$$H(D2)=0.9183$$

$$\text{比例: } 192/384=0.5$$

$$H(D3)=0$$

$$\text{比例: } 64/384=0.1667$$

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

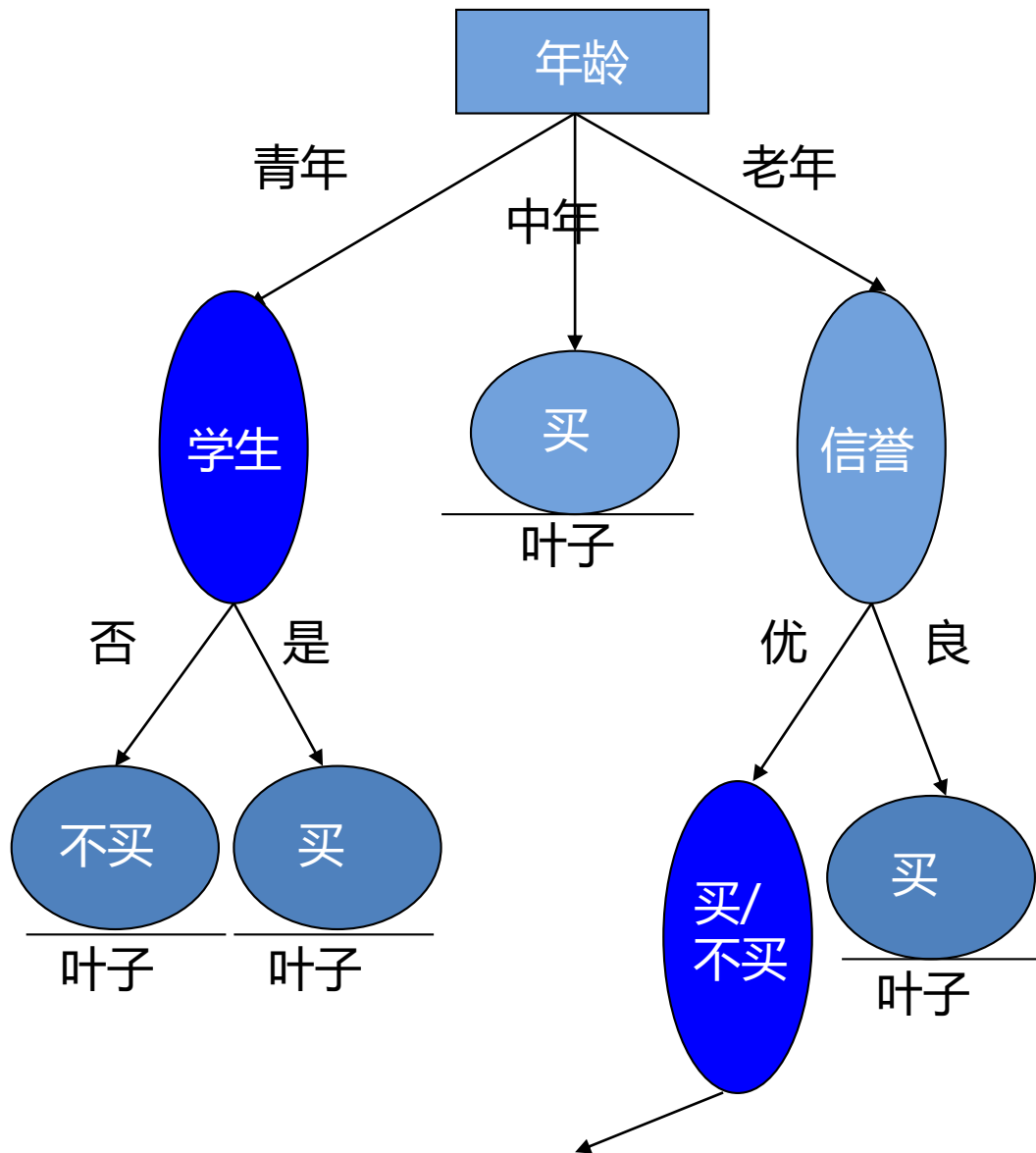
- 平均信息期望（加权总和）：

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

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

# 决策树ID3算法计算步骤

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

---

- 1 决定分类属性;
- 2 对目前的数据表, 建立一个节点N
- 3 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
- 4 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
- 5 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
- 6 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形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

姓名	年龄	收入	学生	信誉	电话	地址	邮编	买计算机
张三	23	4000	是	良	281-322-0328	2714 Ave. M	77388	买
李四	34	2800	否	优	713-239-7830	5606 Holly Cr	78766	买
王二	70	1900	否	优	281-242-3222	2000 Bell Blvd.	70244	不买
赵五	18	900	是	良	281-550-0544	100 Main Street	70244	买
刘兰	34	2500	否	优	713-239-7430	606 Holly Ct	78566	买
杨俊	27	8900	否	优	281-355-7990	233 Rice Blvd.	70388	不买
张毅	38	9500	否	优	281-556-0544	399 Sugar Rd.	78244	买
。 。 。	。 。							
。 。 。								

# 决策树ID3算法-实际使用

- Data cleaning
  - 删除/减少noise,
  - 补填missing values
- Data transformation
  - 数据标准化
  - 数据归纳
  - 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

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
。					
。					
。					

# 决策树ID3算法-小结

---

- ID3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，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。

# 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**过渡拟合**训练样例（过拟合：OverFitting）
- 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。

# 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：

$$C_{\alpha}(T) = \sum_{i=1}^{|T|} N_i H_i(T) + \alpha |T|$$

- 经验熵：

$$H_i(T) = - \sum_k \frac{N_{ik}}{N_i} \log \frac{N_{ik}}{N_i}$$

- 原式第一项：
$$C(T) = \sum_{i=1}^{|T|} N_i H_i(T) = - \sum_{i=1}^{|T|} \sum_{k=1}^K N_{ik} \log \frac{N_{ik}}{N_i}$$

- 则：

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

# 决策树的剪枝

- 树的剪枝算法：
  - 输入：生成算法产生的整数个树 $T$ ，参数 $\alpha$ ；
  - 输出：修剪后的子树 $T_\alpha$
  - (1) 计算每个节点的经验熵
  - (2) 递归地从树的叶节点向上回缩
    - 设一组叶结点回缩到其父结点之前与之后的损失函数分别为： $C_\alpha(T_B)$ 与 $C_\alpha(T_A)$
    - 如果： $C_\alpha(T_B) \leq C_\alpha(T_A)$  则进行剪枝
  - (3) 返回 (2)，直到不能继续为止，得到损失函数最小的子树 $T_\alpha$

# CART树

---

- 分类回归树CART(Classification and Regression Trees)
  - 1984 << Classification and Regression Trees >>
- L.Breiman, J.Friedman, R.Olshen和C.Stone
  - <http://www.stat.berkeley.edu/~breiman/>
  - <http://www-stat.stanford.edu/~jhf/>
  - <http://www-stat.stanford.edu/~olshen/>
- 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的生成

- 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 $R_1, R_2, \dots, R_M$ , 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的生成

- 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：

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

- 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： $\hat{c}_1 = \text{ave}(y_i \mid x_i \in R_1(j, s))$  和  $\hat{c}_2 = \text{ave}(y_i \mid x_i \in R_2(j, s))$
- 再对两个区域重复上述划分，直到满足停止条件。

# CART的生成：最小二乘回归树生成算法

- 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成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

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

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

# CART的生成：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的生成：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：

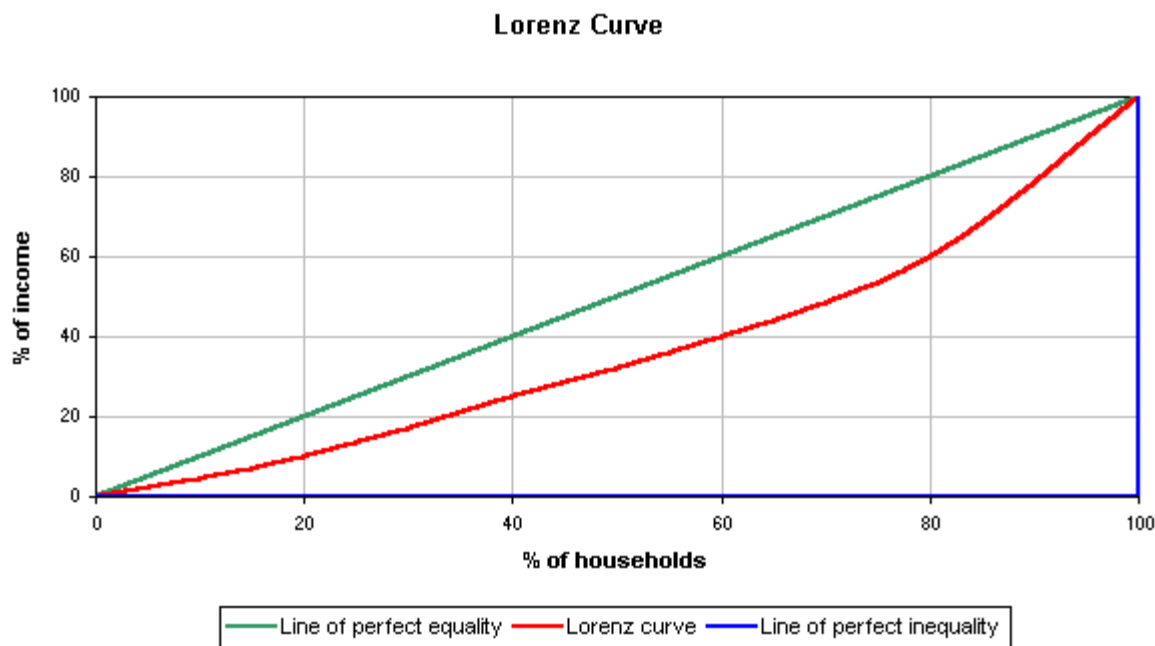
$$\text{Gini}(p) = 2p(1 - p) \quad (5.23)$$

- 对给定的样本集合D，基尼指数

$$\text{Gini}(D) = 1 - \sum_{k=1}^K \left( \frac{|C_k|}{|D|} \right)^2 \quad (5.24)$$

# 基尼指数

- 设图中的实际收入分配曲线（红线）和收入分配绝对平等线（绿线）之间的面积为A，实际收入分配曲线（红线）和收入分配绝对不平等线（蓝线）之间的面积为B，则表示收入与人口之间的比例的基尼系数为 $A/(A+B)$ 。
- 如果A为零，即基尼系数为0，表示收入分配完全平等（红线和绿线重叠）；如果B为零，则系数为1，收入分配绝对不平等（红线和蓝线重叠）。该系数可在0和1之间取任何值。收入分配越趋向平等，劳伦茨曲线的弧度越小（斜度越倾向1:1），基尼系数也越小；反之，收入分配越趋向不平等，劳伦茨曲线的弧度越大，那么基尼系数也越大。



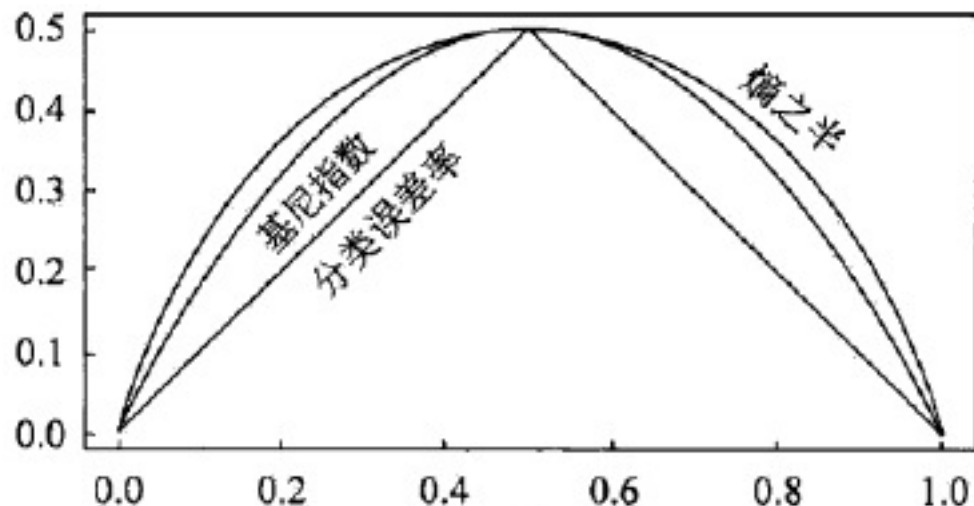
# CART的生成：分类树的生成

- 如果样本集合D根据特征A是否为a被分割成D1和D2，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：

$$\text{Gini}(D, A) = \frac{|D_1|}{|D|} \text{Gini}(D_1) + \frac{|D_2|}{|D|} \text{Gini}(D_2) \quad (5.25)$$



# CART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，递归对内部结点操作
- 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的生成：CART生成算法

---

- 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：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：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，有不等式

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

- 不断增大 $\alpha$ ，某一 $\alpha$ 有

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

- 随着 $\alpha$ 再增大，当

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

- $T_t$ 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：1、剪枝，形成子树序列

- 对T0中每个内部结点t，计算：

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

- 在T0中剪去g(t)最小的Tt，将得到的子树作为T1，同时将最小的g(t)设为a1，T1为区间[a1,a2) 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：2、交叉验证

- 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 $T_0, T_1, \dots, T_n$ 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 $T_1, T_2, \dots, T_n$ 都对应于一个参数 $a_1, a_2, \dots, a_n$ 。
- 所以，当最优子树 $T_k$ 确定时，对应的 $a_k$ 也确定了，即得到最优决策树 $T_a$ 。

# 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 $T_a$ 。

- (1) 设 $k = 0$ ,  $T = T_0$ 。

- (2) 设 $a = +$ 。

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

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

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

- $T_t$ 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) 自上而下地访问内部结点 $t$ , 如果有 $g(t) = \alpha$ , 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 $T$ 不是由根结点单独构成的树, 则回到步骤 (4)。

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

