

第5章 决策树



主要思想

基于特征对实例进行分类，包括特征选择、决策树生成和决策树剪枝

主要内容

- 决策树模型与学习
- 特征选择
- 决策树的生成
- 决策树的剪枝
- C A R T 算法



5.1 决策树模型与学习

【引例】相亲问题

女儿：多大年纪了？

母亲：26。

女儿：长的帅不帅？

母亲：挺帅的。

女儿：收入高不？

母亲：不算很高，中等情况。

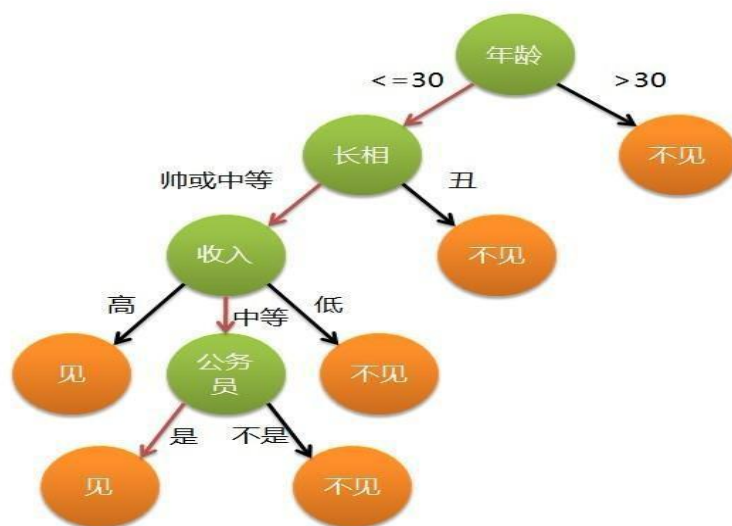
女儿：是公务员不？

母亲：是，在税务局上班呢。

女儿：那好，我去见见。

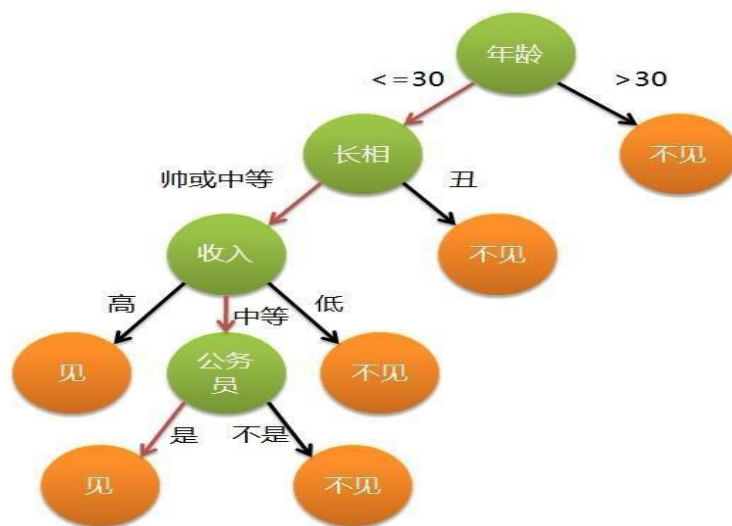
1、决策树模型

【定义】它是一种描述对**实例样本**进行分类的**树形结构**，由结点（node）和有向边（directed edge）组成。



结点有两种类型：内部结点(internal node)和叶结点(leaf node)，内部结点表示一个特征，叶结点表示一个类别(分类结果)

- 叶结点 (leaf node) : 同一分类结果
- 内部结点 (internal node) : 多个分类结果
- 内部结点层次 : 特征重要程度





5.2 特征选择

1、选择哪个特征好呢？

特征分类能力强

||

数据子集中样本几乎属于同一类别

||

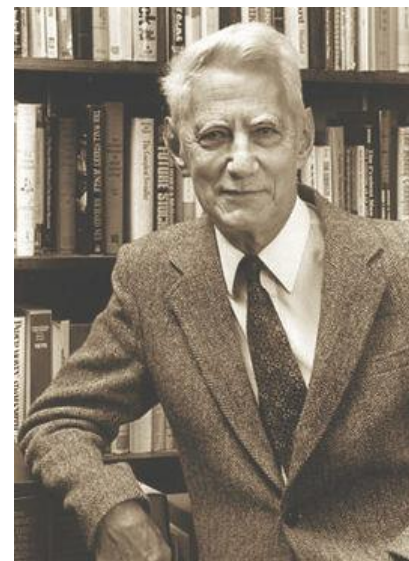
数据子集中样本纯度高


2、信息熵

我们常说信息很多，或信息很少，但却很难说清楚信息到底有多少？

比如一本50多万字的《史记》有多少信息量？或一套莎士比亚全集有多少信息量？

这个问题几千年来都没有人给出很好的解答，直到1948年，香农(Claude Shannon)在他著名的论文“通信的数学原理”中提出了信息熵的概念，才解决了信息的度量问题，并且量化出信息的作用。






假如我错过了一个有32支球队参加的足球赛，赛后我问一个知道比赛结果的观众“哪支球队是冠军”？他不愿意直接告诉我，而让我猜，每猜一次，他要收一元钱后才肯告诉我是否猜对，那我需要付多少钱才能知道谁是冠军呢？

把球队编号，从1到32，然后问“冠军球队在1-16号中吗？”，假如他告诉我猜对了，我就接着问“冠军在1-8号中吗？”，假如他说猜错了，那我就知道冠军在9-16号中。这样只要5次，我就能知道哪支球队是冠军。

当然，香农不是用钱，而是用比特(bit)来度量信息量，在上例中，这条消息的信息量是5比特。



实际上可能不需要5次就能猜出谁是冠军，因为一些强队得冠的可能性更高，因此第一次猜测时可以把少数几支强队分成一组，其它球队分成另一组，然后猜冠军球队是否在那几支强队中，这样，也许三次或四次就能猜出结果。因此当每支球队夺冠的可能性(概率)不等时，这条信息的信息量比5比特少。

香农指出，它的准确信息量应该是：

$$H=-(p_1\log p_1+p_2\log p_2+\dots+p_{32}\log p_{32}).$$

其中 p_1, p_2, \dots, p_{32} 是这32支球队夺冠概率。香农把它称作信息熵，单位为比特

设X是一个取值离散的随机变量，其概率分布为 $P(X=x_i)=p_i$ ， $i=1,2,\dots,n$ ，则随机变量X的信息熵定义为：

$$H(X) = -\sum_{i=1}^n p_i \log_2 p_i$$

变量的不确定性越大，熵也就越大，把它搞清楚所需要的信息量也就越大

使用熵衡量数据纯度

假设有一个数据集D，其中只有两个类，一个是正例类，一个是负例类
计算D中正例类和负例类在三种不同的组分下熵的变化情况。

(1) D中包含有50%的正例和50%的负例。

$$H(D) = -0.5 * \log_2 0.5 - 0.5 * \log_2 0.5 = 1$$

(2) D中包含有20%的正例和80%的负例。

$$H(D) = -0.2 * \log_2 0.2 - 0.8 * \log_2 0.8 = 0.722$$

(3) D中包含有100%的正例和0%的负例。

$$H(D) = -1 * \log_2 1 - 0 * \log_2 0 = 0$$

可以看到一个趋势，当数据变得越来越“纯”时，熵的值变得越来越小。

当D中正反例所占比例相同时，熵取最大值。

当D 中所有数据都只属于一个类时，熵得到最小值。

因此熵可以作为数据纯净度或混乱度的衡量指标。这正是决策树学习中需要的。

3、条件熵 $H(Y|X)$

在随机变量 X 给定条件下，随机变量 Y 的**条件概率分布的熵**对随机变量 X 的数学期望

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

称为随机变量 Y 的条件熵 $H(Y|X)$ 。

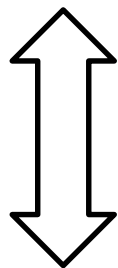
4、信息增益 $g(D,A)$

训练数据集D的信息熵 $H(D)$ ，与特征A给定条件下 D的条件熵 $H(D|A)$ 之差：

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

称为特征A对训练数据集D的信息增益。

选择具有最高信息增益 $g(A)$ 的属性A作为分裂属性



按照能做“最佳分类”的属性A划分，使完成样本分类需要的信息量 $H(D|A)$ 最小

5、信息增益算法

(1) 计算数据集D的信息熵H(D)

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

|D| 表示样本的总容量。

设有K个类 C_k , $k=1, 2, \dots, n$.

| C_k | 属于 C_k 的样本个数。

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) 计算特征A对数据集D的条件熵 $H(D|A)$

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

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，将数据集D划分为n个子集， D_1, D_2, \dots, D_n 。

$|D_i|$ 为 D_i 的样本个数。

D_{ik} 为子集 D_i 中属于类 C_k 的样本的集合， $D_{ik} = D_i \cap C_k$ 。

$|D_{ik}|$ 为 D_{ik} 的样本个数。

(3) 计算信息增益

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

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

【例】图中所示的数据集D，根据信息增益准则，选择最优特征。

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

首先计算决策属性的信息熵

决策属性“买计算机？”
该属性分两类：买/不买

$$|C_1| \text{ (买)} = 641$$

$$|C_2| \text{ (不买)} = 383$$

$$|D| = 1024$$

$$H(D) = - \sum_{k=1}^2 \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|}$$

$$= 0.9537$$

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

然后计算各个特征对数据集D的信息增益，分别以A1， A2， A3， A4表示年龄、收入、学生、信誉4个特征，则：

$$g(D, A) = H(D) - H(D \mid A)$$

$$H(D \mid A) = \sum_{i=1}^n \frac{|D_i|}{|D|} H(D_i)$$

$$H(D_i) = \sum_{k=1}^K \frac{|D_{ik}|}{|D_i|} \log_2 \frac{|D_{ik}|}{|D_i|}$$

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

(1) 设数据集D按照年龄特征A1划分，取值为青、中和老的样本子集分别为D1，D2和D3。

青年数据子集D1中：

$$|D_{11}|(\text{买})=128, |D_{12}|(\text{不买})=256$$

$$|D_1|=384$$

青年的信息熵：

$$H(D_1) = \sum_{k=1}^2 \frac{|D_{1k}|}{|D_1|} \log_2 \frac{|D_{1k}|}{|D_1|}$$

$$= 0.9183$$

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

中年数据子集D2中：
 $|D21|(\text{买})=256$ ， $|D22|(\text{不买})= 0$
 $|D2|=256$ ， $H(D2)=0$

老年数据子集D3中：
 $|D31|(\text{买})=257$ ， $|D32|(\text{不买})=127$
 $|D3|=252$ ， $H(D3)=0.9157$

年龄的信息增益：

$$g(D, A_1) = H(D) - \sum_{i=1}^n \frac{|D_i|}{|D|} H(D_i)$$

$$= 0.9537 - (\frac{384}{1024} * 0.9183 + \frac{252}{1024} * 0.9157)$$

$$= 0.2660$$

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

年龄的信息增益=**0.2660**

(2) 计算收入的熵

收入共分3个数据子集：高、中、低

收入信息增益=0.0176

(3) 计算学生的熵

学生共分2个数据子集：学生、非学

生

学生信息增益=0.1726

(4) 计算信誉的熵

信誉分2个数据子集：良好，优秀

信誉信息增益=0.0453



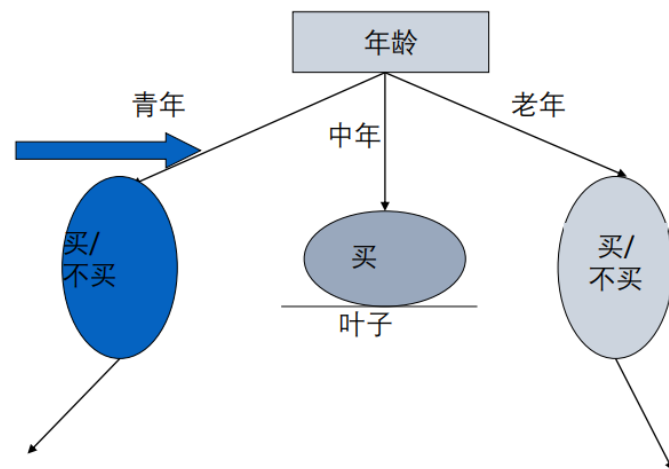
5.3决策树的生成

1、ID3决策树算法

ID3算法是一种经典的决策树学习算法，由Quinlan于1979年提出。是决策树学习方法中最具影响和最为典型的算法。

基本思想是，以信息增益为度量，用于决策树节点的特征选择，每次优先选取信息增益最大的特征，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

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



2、C4.5算法对ID3的改进

改进1：用信息增益率代替信息增益来选择属性

改进2：连续值的处理

改进3：能处理属性值缺失的情况

改进4：在决策树构造完成之后进行剪枝

<i>Tid</i>	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	—	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

改进1：信息增益的问题

信息增益度量偏向于对取值较多的属性进行测试，即它倾向于选择n较大的属性A

特征A对训练数据集D的信息增益比 $g_k(D, A)$ ：

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

C4.5使用分裂信息(split information)将信息增益规范化,选择具有最大信息增益率的属性作为分裂属性

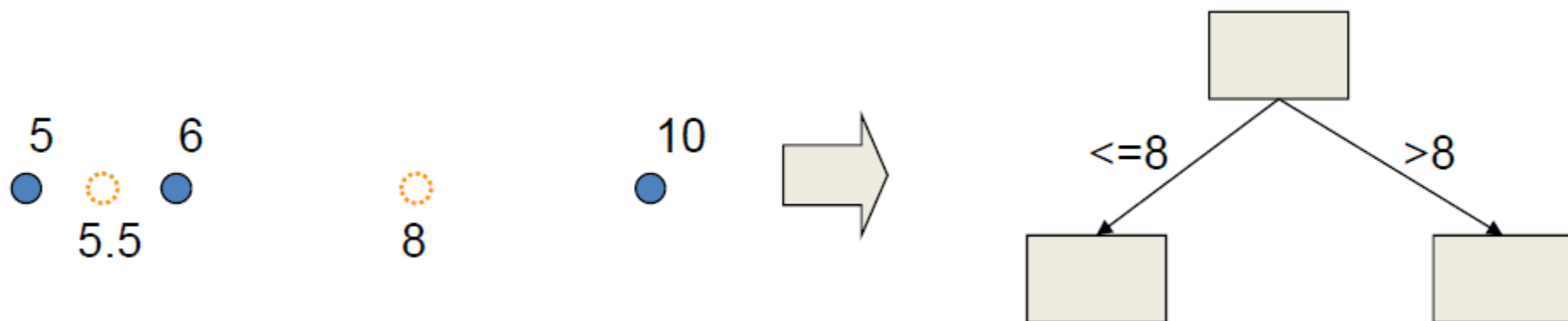
通常先从候选划分属性中找出信息增益高于平均水平的属性，再从中选择增益率最高的属性进行划分。

改进2：连续值处理

对于连续值属性，按属性值大小从小到大排序，取每对相邻值的中点作为可能的分裂点split_point。

假设一连续值属性共有N个不同的属性值，则可找到N-1个可能的分裂点。

检查每个可能分裂点，取能使得**信息增益最大**的分裂点，将D分裂成
D1: $A \leq \text{split_point}$ 和 D2: $A > \text{split_point}$ (一个分裂点，二分法，二叉树)



使用二元划分

- 将记录进行排序
- 从两个相邻的排过序的属性值之间选择中间值作为划分点
- 计算每个候选点的信息增益值
- 选择修正后信息增益最大的分裂点作为该特征的最佳分裂点
- 计算最佳分裂点的信息增益率作为特征的信息增益率
- 时间复杂度为 $n\log n$

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

$$H(D) = -0.3 \log_2 0.3 - 0.7 \log_2 0.7 = 0.881$$

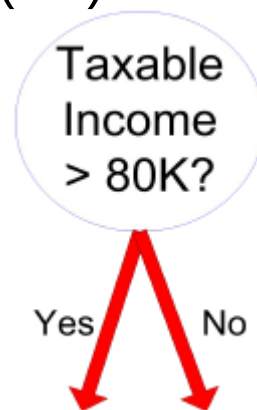
以“ 80” 为分裂点，计算信息增益

$$H(D1)=0 \quad H(D2)=-\frac{3}{7}\log_2 \frac{3}{7} - \frac{4}{7}\log_2 \frac{4}{7}$$

$$g(D,A) = 0.881 - 0.3H(D1) - 0.7H(D2) \\ = 0.881 - 0.689 = 0.192$$

$$H_A(D) = -0.3 \log_2 0.3 - 0.7 \log_2 0.7$$

$$g_k(D,A) = 0.192/0.881$$

[illegible]


改进3：C4.5中缺失值的处理

在某些情况下，可供使用的数据可能缺少某些属性的值，例如

天气	湿度	有雨？	去玩？
晴	70	有	玩
晴	90	有	不玩
晴	85	无	不玩
晴	95	无	不玩
晴	70	无	玩
缺失	90	有	玩
多云	78	无	玩
多云	65	有	玩
多云	75	无	玩
雨	80	有	不玩
雨	70	有	不玩
雨	80	无	玩
雨	80	无	玩
雨	96	无	玩

简单的办法是赋予它该属性最常见的值，例如将“晴”或“雨”赋予第6个实例的天气属性

更复杂的策略是为A的每个可能值赋予一个概率



建树过程（学习过程）

– 选定训练样本实例有缺失值，如何知道要将其分配到哪个分支？

分类过程（测试过程或者工作过程）

– 待分类实例有缺失值，如何测试该实例属于哪个分支？

(天气=缺失，温度=90，有雨=是...)

(1) 学习过程

计算信息增益率时，将缺失的属性值当作一个正常值进行计算，本例中，当作天气有四个值，分别是晴, 多云, 雨, ?，再计算信息增益率。

天气	湿度	有雨?	去玩?
晴	70	有	玩
晴	90	有	不玩
晴	85	无	不玩
晴	95	无	不玩
晴	70	无	玩
缺失	90	有	玩
多云	78	无	玩
多云	65	有	玩
多云	75	无	玩
雨	80	有	不玩
雨	70	有	不玩
雨	80	无	玩
雨	80	无	玩
雨	96	无	玩

$$\begin{aligned} H(D|\text{天气}) &= -5/14 \times \log(5/14) \\ &\quad - 3/14 \times \log(3/14) \\ &\quad - 5/14 \times \log(5/14) \\ &\quad - 1/14 \times \log(1/14) \\ &= 1.809 \text{ bits} \end{aligned}$$

$$\begin{aligned} g_k(D, \text{天气}) &= g(D, \text{天气}) / H(D|\text{天气}) \\ &= 0.199 / 1.809 \end{aligned}$$

分裂时，将属性值缺失的实例分配给所有分支，但是带一个权重

T1: (天气=晴)

湿度	有雨	玩?	权重
70	有	玩	1
90	有	不玩	1
85	无	不玩	1
95	无	不玩	1
70	无	玩	1
90	有	玩	5/13

T1: (天气=多云)

湿度	有雨	玩?	权重
90	有	玩	3/13
78	无	玩	1
65	有	玩	1
75	无	玩	1

T1: (天气=雨)

湿度	有雨	玩?	权重
80	有	不玩	1
70	有	不玩	1
80	无	玩	1
80	无	玩	1
96	无	玩	1
90	有	玩	5/13

本例14个实例中共13个实例天气属性值未缺失：

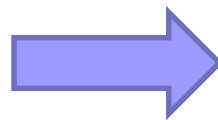
其中5个实例的天气属性为“晴”，3个实例的天气属性为“多云”，5个实例的天气属性为“雨”

本例14个实例中共1个实例天气属性值缺失，因此估算出天气属性值缺失的第6个实例：

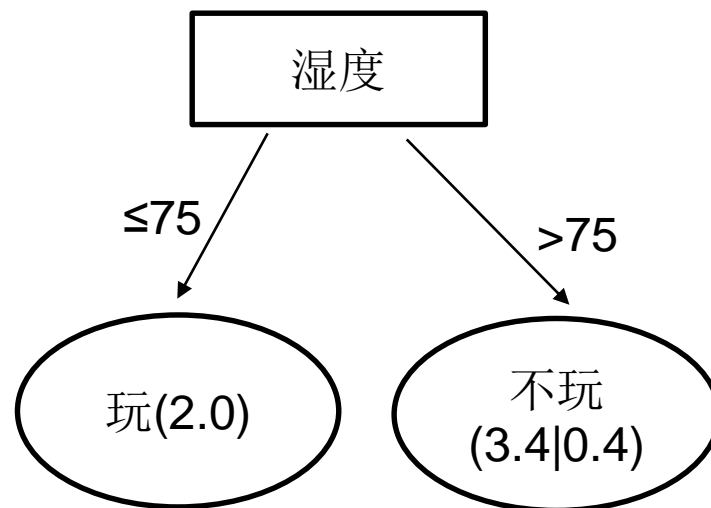
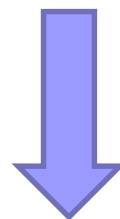
天气是晴的概率是5/13，天气是多云的概率是3/13，天气是雨的概率是5/13

T1: (天气=晴)

湿度	有雨	玩?	权重
70	有	玩	1
90	有	不玩	1
85	无	不玩	1
95	无	不玩	1
70	无	玩	1
90	有	玩	5/13=0.4



湿度 ≤ 75 2玩, 0不玩
湿度 > 75 5/13玩, 3不玩



叶节点以 (N/E) 的形式定义，
其中 N 为到达该叶节点的实例数，
E 为其中属于其它分类的实例数。
例如，**不玩(3.4/0.4)** 表示3.4个实例
到达“不玩”节点，其中0.4个实例
不属于“不玩”

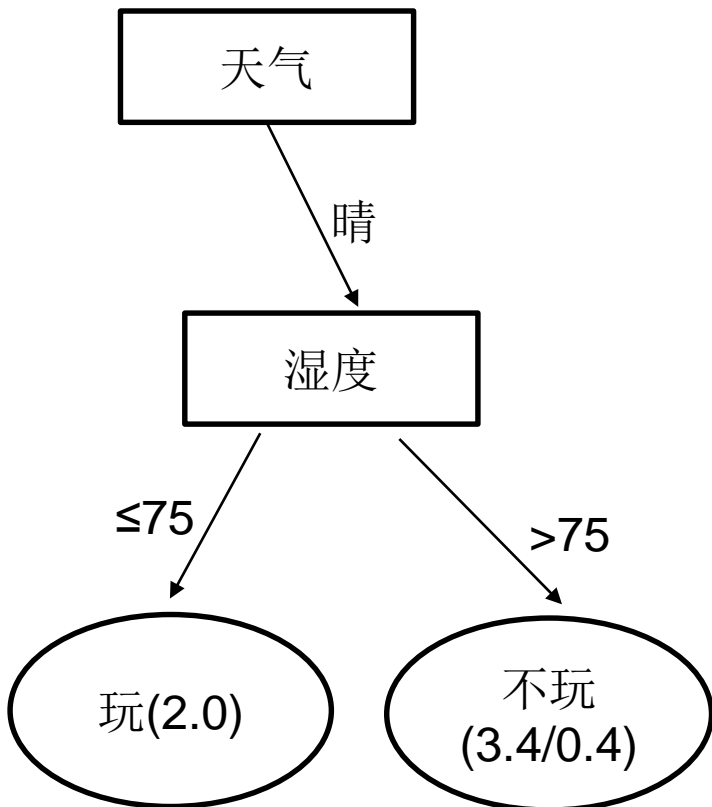
(2) 分类过程

(天气=晴, 湿度=缺失, 如何判断是否玩?)

对于任一实例,
湿度 ≤ 75 的可能性是 $2.0/(2.0 + 3.4)$
湿度 > 75 的可能性是 $3.4/(2.0 + 3.4)$

当湿度 ≤ 75 时,
分类为玩的可能性 = 100%
分类为不玩的可能性 = 0
当湿度 > 75 时,
分类为玩的可能性 = $0.4/3.4 = 12\%$
分类为不玩的可能性 = $3/3.4 = 88\%$

最终分类的概率分布为:
玩 = $2.0/5.4 \times 100\% + 3.4/5.4 \times 12\% = 44\%$
不玩 = $3.4/5.4 \times 88\% = 56\%$



改进4：学习过程中的过度拟合

上述的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，当训练样本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度拟合训练样例 (过度拟合: Over fitting)

过度拟合产生的原因：训练样本中有噪声，训练样例太小等

决策树剪枝

How

预剪枝(prepruning):

在完全正确分类训练集之前就停止树的生长。

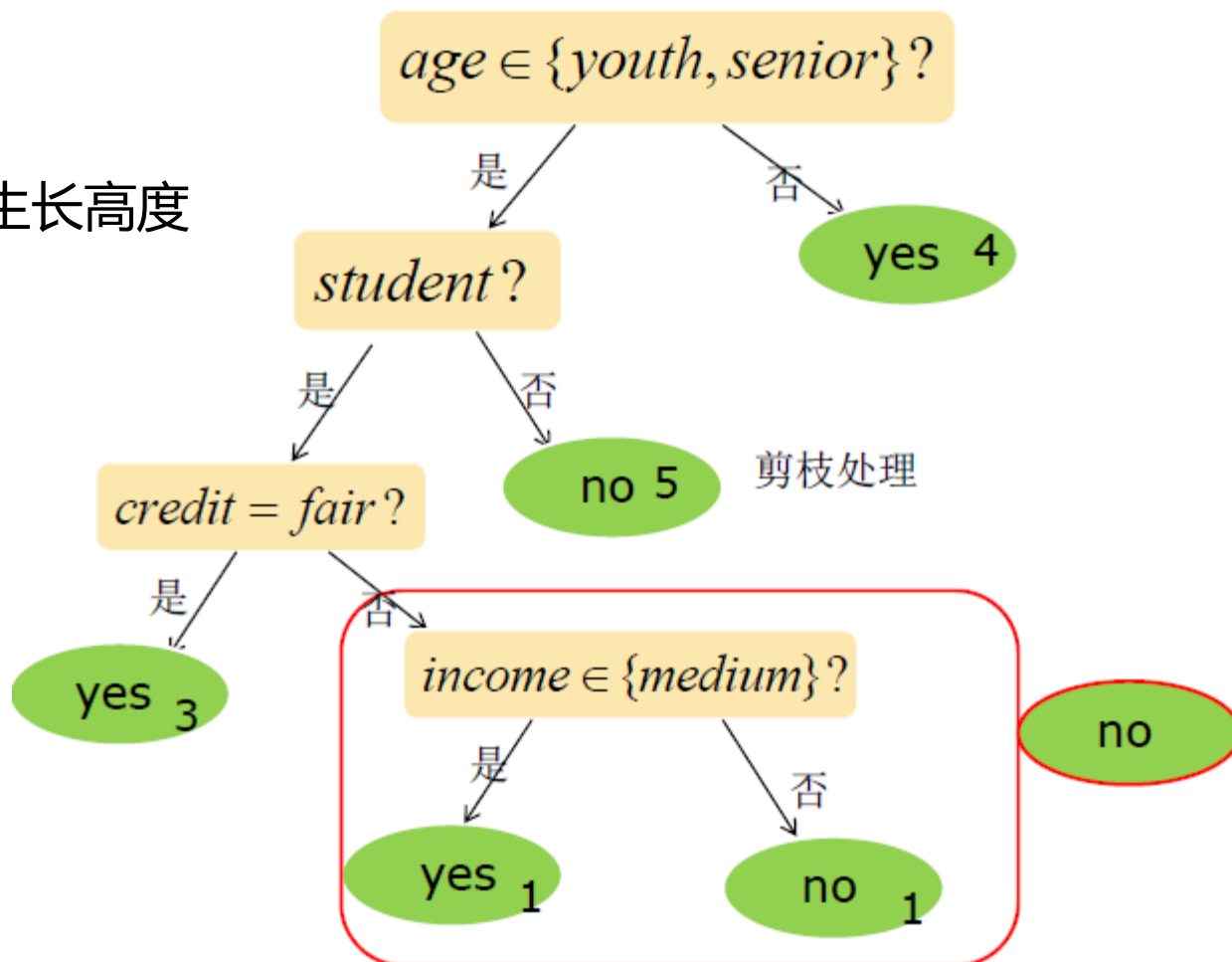
后剪枝(postpruning):

由“完全生长”的树剪去子树。

预剪枝

最直接的方法：
事先限定树的最大生长高度

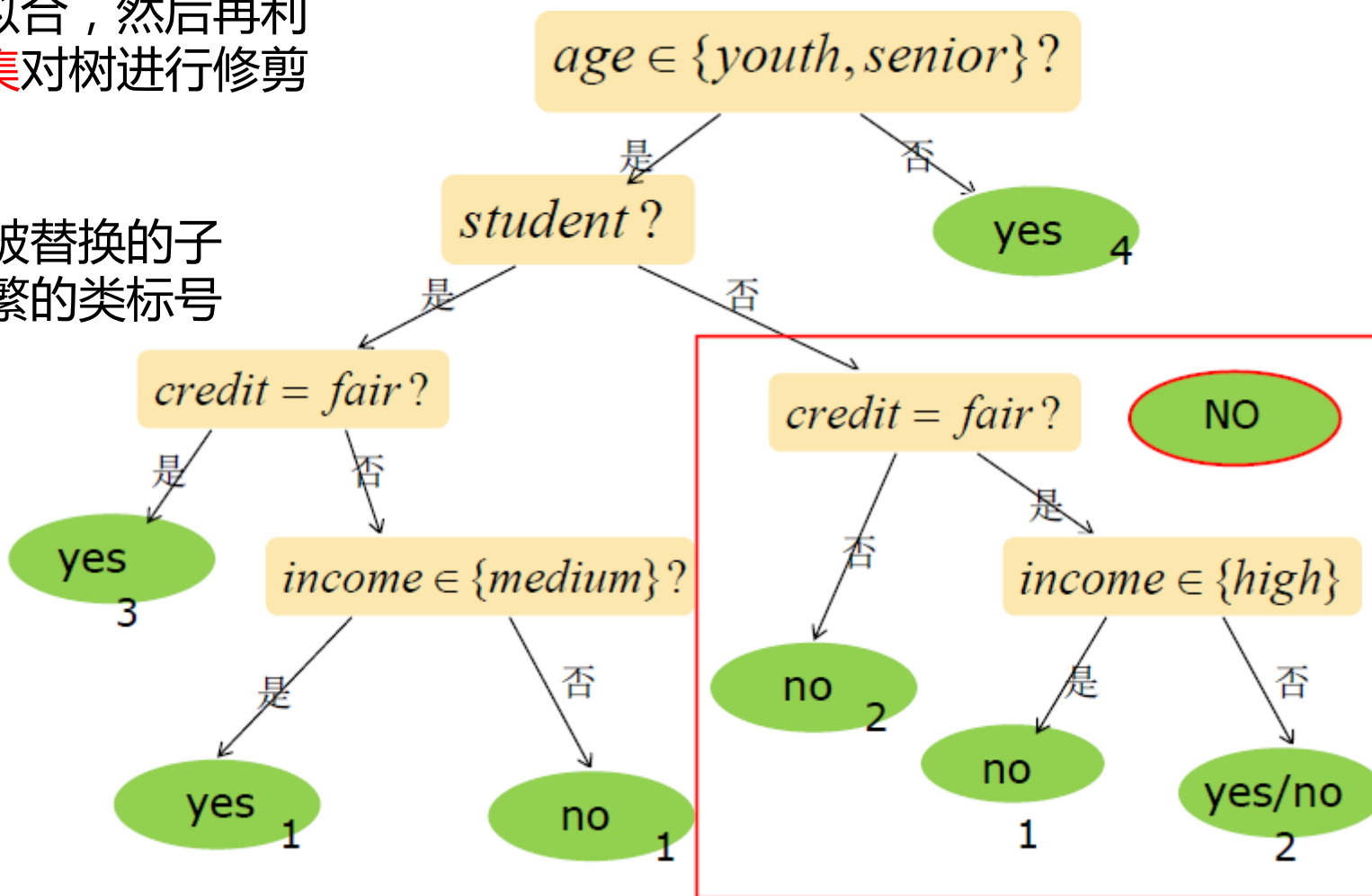
如果设为3，
则如图剪枝



后剪枝

训练过程中允许对数据的过度拟合，然后再利用测试集对树进行修剪

树叶用被替换的子树最频繁的类标号



后剪枝

在测试集上定义损失函数 C ，我们的目标是通过剪枝使得在测试集上损失函数下降。

1. 自底向上的遍历每一个非叶节点(除了根节点)，将当前的非叶节点从树中减去，其下所有的叶节点合并成一个节点，代替原来被剪掉的节点。
2. 计算剪去节点前后的损失函数，如果剪去节点之后损失函数变小了，则说明该节点是可以剪去的，并将其剪去；如果发现损失函数并没有减少，说明该节点不可剪去，则将树还原成未剪去之前的状态。
3. 重复上述过程，直到所有的非叶节点(除了根节点)都被尝试了。

损失函数

设树T的叶结点个数为 $|T|$ ， t 是树的叶结点，该结点有 N_t 个样本点，其中 k 类的样点为 N_{tk} ， $k=1,2,\dots,K$ ，决策树学习的损失函数为：

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

$H_t(T)$ 为叶结点 t 上的信息熵， $\alpha \geq 0$ 为参数

$$H_t(T) = - \sum_{k=1}^K \frac{N_{tk}}{N_t} \log_2 \frac{N_{tk}}{N_t}$$


第1项记为 $C(T) = \sum_{t=1}^{|T|} N_t H_t(T) = - \sum_{t=1}^{|T|} \sum_{k=1}^K N_{tk} \log_2 \frac{N_{tk}}{N_t}$

则 $C_{\alpha}(T) = C(T) + \alpha |T|$

其中 $C(T)$ 表示模型对训练数据的预测误差，即模型和训练数据的拟合程度。
 $|T|$ 表示模型复杂度。



5.4 CART算法



CART，又名分类回归树，是在ID3的基础上进行优化的决策树，学习CART记住以下几个关键点：

(1) CART既能是分类树，又能是回归树；

(2) 当CART是分类树时，采用GINI值作为节点分裂的依据；
当CART是回归树时，采用样本的最小方差作为节点分裂的依据；

(3) CART是一棵二叉树。

分类树的作用是通过一个对象的特征来预测该对象所属的类别，而回归树的目的是根据一个对象的信息预测该对象的属性，并以数值表示。

看电视时间	婚姻情况	职业	年龄
3	未婚	学生	12
4	未婚	学生	18
2	已婚	老师	26
5	已婚	上班族	47
2.5	已婚	上班族	36
3.5	未婚	老师	29
4	已婚	学生	21

CART既是分类树，又是回归树，如上表所示，如果我们想预测一个人是否已婚，那么构建的CART将是分类树；如果想预测一个人的年龄，那么构建的将是回归树。

分类树和回归树是怎么做决策的？假设我们构建了两棵决策树分别预测用户是否已婚和实际的年龄，如图1和图2所示：

看电视时间	婚姻情况	职业	年龄
3	未婚	学生	12
4	未婚	学生	18
2	已婚	老师	26
5	已婚	上班族	47
2.5	已婚	上班族	36
3.5	未婚	老师	29
4	已婚	学生	21

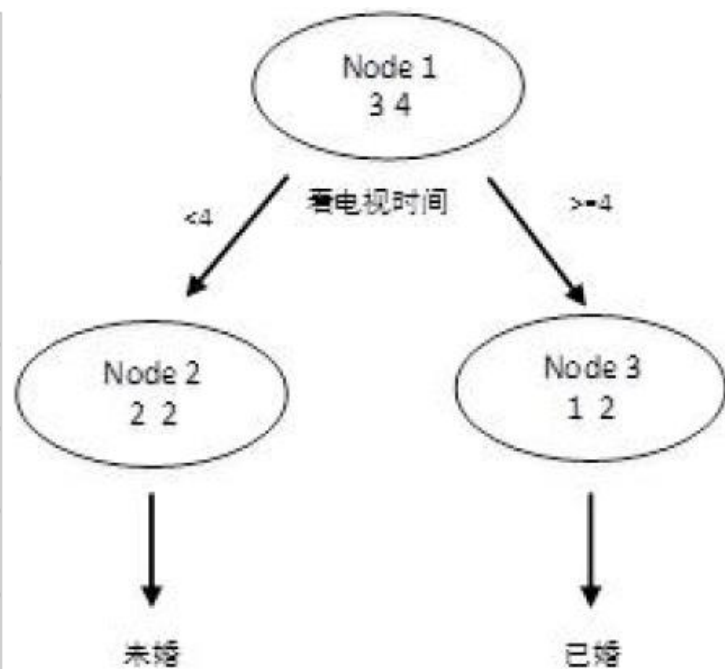


图1表示一棵分类树，其叶子节点的输出结果为一个实际的类别，在这个例子里是婚姻的情况（已婚或者未婚），选择叶子节点中数量占比最大的类别作为输出的类别；

分类树和回归树是怎么做决策的？假设我们构建了两棵决策树分别预测用户是否已婚和实际的年龄，如图1和图2所示：

看电视时间	婚姻情况	职业	年龄
3	未婚	学生	12
4	未婚	学生	18
2	已婚	老师	26
5	已婚	上班族	47
2.5	已婚	上班族	36
3.5	未婚	老师	29
4	已婚	学生	21

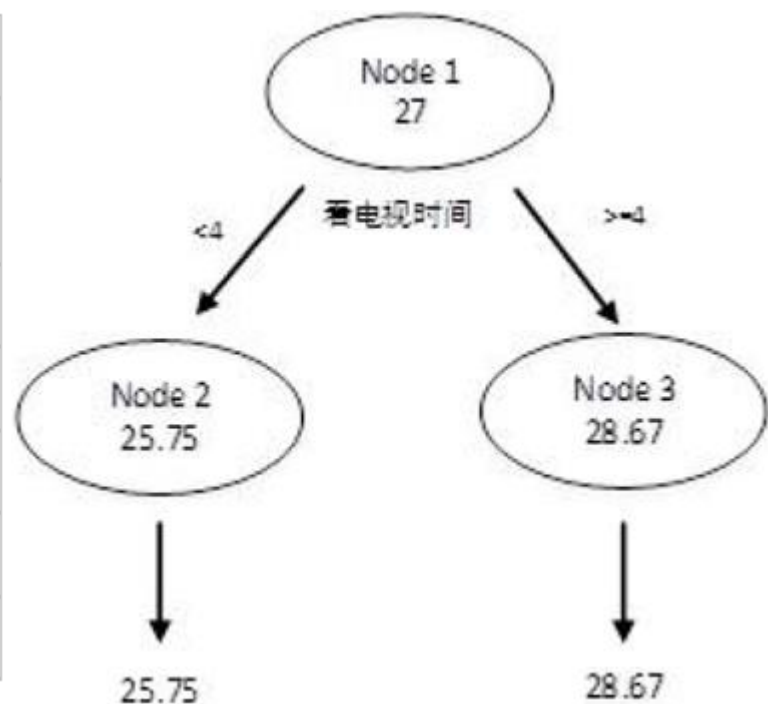


图2是一棵回归树，预测用户的实际年龄，是一个具体的输出值。怎样得到这个输出值？一般情况下选择使用中值、、平均值或者众数进行表示，图2使用节点年龄数据的平均值作为输出值。

分类树---GINI值

分类问题中，假设有k个类，样本点属于k的概率 P_k ，则概率分布的基尼值

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

对给定的样本集合D，基尼值：

$$Gini(D) = 1 - \sum_{k=1}^K \left(\frac{|C_k|}{|D|} \right)^2 \quad \text{节点纯度越大，GINI值越小}$$

基尼指数：

$$Gini_index(D, A) = \sum_{i=1}^n \frac{|D_i|}{|D|} Gini(D_i)$$

在候选特征中，选择基尼指数最小的特征作为划分特征。

回归树---回归方差

回归方差计算公式

$$\sigma = \sqrt{\sum_{x_i \in R_m} (x_i - u)^2} = \sqrt{\sum_{x_i \in R_m} x_i^2 - nu^2}$$

如果两类数量相同，则方差越大，表示该节点的数据越分散，预测的效果就越差。

如果一个节点的所有数据都相同，那么方差就为0，此时可以很肯定得认为该节点的输出值；

如果节点的数据相差很大，那么输出的值有很大的可能与实际值相差较大。

以属性“职业”为例，该属性有三种划分的方案，分别计算三种划分方案的子节点GINI值或者样本方差，选择最优的划分方法。

第一种划分方法：{ “学生” }、{ “老师” 、 “上班族” }

看电视时间	是否已婚	职业	年龄
3	否	学生	12
4	否	学生	18
2	是	老师	26
5	是	上班族	47
2.5	是	上班族	36
3.5	否	老师	29
4	是	学生	21

看电视时间	是否已婚	职业	年龄
3	否	学生	12
4	否	学生	18
4	是	学生	21

看电视时间	是否已婚	职业	年龄
2	是	老师	26
5	是	上班族	47
2.5	是	上班族	36
3.5	否	老师	29

预测是否已婚（分类）：

$$Gain = \sum_{i \in I} p_i Gini_i = \frac{3}{7} (1 - ((\frac{2}{3})^2 + (\frac{1}{3})^2)) + \frac{4}{7} (1 - ((\frac{3}{4})^2 + (\frac{1}{4})^2)) = 0.4$$

预测年龄（回归）：

$$Gain = \sum_{i \in I} \sigma_i = \sqrt{12^2 + 18^2 + 21^2 - 3 \cdot 17^2} + \sqrt{26^2 + 47^2 + 36^2 + 29^2 - 4 \cdot 32.5^2} = 34.71$$

1、分类树与回归树的构建

无论是分类树还是回归树，CART都要选择使子节点的GINI值或者回归方差最小的属性作为分裂的方案。即最小化（分类树）。

如何分裂成一个二叉树？

节点的分裂分为两种情况，连续型的数据和离散型的数据。

对于离散型属性，理论上有多少个离散值就应该分裂成多少个节点。但CART是一棵二叉树，每一次分裂只会产生两个节点。只要将其中一个离散值独立作为一个节点，其他的离散值生成另外一个节点即可。这种分裂方案有多少个离散值就有多少种划分的方法，分别计算每种划分方法的基尼值或者样本方差确定最优的方法。

第二种划分方法：{ “老师” }、{ “学生”、“上班族” }

看电视时间	是否已婚	职业	年龄
3	否	学生	12
4	否	学生	18
2	是	老师	26
5	是	上班族	47
2.5	是	上班族	36
3.5	否	老师	29
4	是	学生	21

看电视时间	是否已婚	职业	年龄
2	是	老师	26
3.5	否	老师	29

看电视时间	是否已婚	职业	年龄
3	否	学生	12
4	否	学生	18
5	是	上班族	47
2.5	是	上班族	36
4	是	学生	21

预测是否已婚（分类）：

$$\begin{aligned}
 Gain &= \sum_{i \in I} p_i Gini_i = \frac{2}{7} (1 - ((\frac{1}{2})^2 + (\frac{1}{2})^2)) \\
 &\quad + \frac{5}{7} (1 - ((\frac{3}{5})^2 + (\frac{2}{5})^2)) = 0.49
 \end{aligned}$$

预测年龄（回归）：

$$\begin{aligned}
 Gain &= \sum_{i \in I} \sigma_i = \sqrt{26^2 + 29^2 - 2 \bullet 27.5^2} \\
 &\quad + \sqrt{12^2 + 18^2 + 47^2 + 36^2 + 21^2 - 5 \bullet 26.8^2} \\
 &= 16.36
 \end{aligned}$$

第三种划分方法：{ “上班族” }、{ “学生”、“老师” }

看电视时间 _i	是否已婚 _i	职业 _i	年龄 _i
3	否	学生	12
4	否	学生	18
2	是	老师	26
5	是	上班族	47
2.5	是	上班族	36
3.5	否	老师	29
4	是	学生	21

看电视时间 _i	是否已婚 _i	职业 _i	年龄 _i
5	是	上班族	47
2.5	是	上班族	36

看电视时间 _i	是否已婚 _i	职业 _i	年龄 _i
3	否	学生	12
4	否	学生	18
2	是	老师	26
3.5	否	老师	29
4	是	学生	21

预测是否已婚（分类）：

$$Gain = \sum_{i \in I} p_i Gini_i = \frac{2}{7} (1 - 1) + \frac{5}{7} (1 - ((\frac{3}{5})^2 + (\frac{2}{5})^2)) = 0.34$$

预测年龄（回归）：

$$Gain = \sum_{i \in I} \sigma_i = \sqrt{47^2 + 36^2 - 2 \bullet 41.5^2} + \sqrt{12^2 + 18^2 + 26^2 + 29^2 + 21^2 - 5 \bullet 21.2^2} = 21.14$$

综上，如果想预测是否已婚，则选择{ “上班族” }、{ “学生”、“老师” }的划分方法，如果想预测年龄，则选择{ “老师” }、{ “学生”、“上班族” }的划分方法。

2、CART剪枝

CART采用CCP（代价复杂度）剪枝方法。代价复杂度选择节点**表面误差率增益**值最小的非叶子节点，删除该非叶子节点的左右子节点，若有多个非叶子节点的表面误差率增益值同样小，则选择非叶子节点中子节点数最多的非叶子节点进行剪枝。

令决策树的非叶子节点为 $T = \{T_1, T_2, \dots, T_n\}$

a) 计算所有非叶子节点的表面误差率增益值 $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$

b) 选择表面误差率增益值 α_i 最小的非叶子节点 T_i （若多个非叶子节点具有同样小的表面误差率增益值，选择节点数最多的非叶子节点）

c) 对 T_i 进行剪枝

表面误差率增益
$$g(t) = \frac{C(t) - C(T_t)}{|T_t| - 1}$$

t 为整体树 T_0 的任意内部结点， $C(t)$ 表示以 t 为单结点树的损失函数。
 $C(T_t)$ 表示以 T_t 为根结点子树 T_t 的损失函数。

决策树的总结

决策树学习采用的是自顶向下的递归方法。

其基本思想是以信息熵为度量构造一棵熵值下降最快的树，到叶子节点处的熵值为零，此时每个叶节点中的实例都属于同一类。

构造一棵决策树要解决四个问题：

- (1) 收集待分类的数据，这些数据的所有属性应该是完全标注的。
- (2) 设计分类原则，即数据的哪些属性可以被用来分类，以及如何将该属性量化。
- (3) 分类原则的选择，即在众多分类准则中，每一步选择哪一准则使最终的树更令人满意。
- (4) 设计分类停止条件，实际应用中数据的属性很多，真正有分类意义的属性往往是有限几个，因此在必要的时候应该停止数据集分裂：
 - 该节点包含的数据太少不足以分裂，
 - 继续分裂数据集对树生成的目标(例如**ID3**中的熵下降准则)没有贡献
 - 树的深度过大不宜再分。

通用的决策树分裂目标是整棵树的熵总量最小，每一步分裂时，选择使熵减小最大的准则，这种方案使最具有分类潜力的准则最先被提取出来。



决策树特点

决策树有很多的优点，可解释性、计算快捷、缺失值的处理、对于多值名义变量不需要建立哑变量、对输入变量异常值稳健。

对数据的要求

进行分析时，决策树对变量的量纲的差异、离群值的存在以及有偏分布不太敏感，也就是说对数据准备要求不高。

当每一类的训练样本数较小时，决策树是容易出错的，有好多分支的树或者每个节点有太多枝的树最有可能这样，决策树对输出结果的密度很敏感；



5.5 随机森林

集成学习

通过**将多个单个学习器集成/组合**在一起，使它们共同完成学习任务，有时也被称为“多分类器系统（multi-classifier system）”、基于委员会的学习（Committee-based learning）。

集成学习方法的一般过程:

- (1) 令 D 表示原始训练数据集， k 表示基分类器的个数， Z 表示测试数据集
- (2) for $i=1$ to k do
- (3) 由 D 创建训练集 D_i
- (4) 由 D 创建基分类器 C_i
- (5) end for
- (6) for 每一个测试样本 do
- (7) $C^*(x) = \text{Vote}(C_1(x), C_2(x), \dots, C_k(x))$
- (8) end for

基础集成技术

假设一名电影导演，依据一个非常重要且有趣的话题创作了一部短片。现在想在公开发布前获得影片的初步反馈（评级）。邀请5位同事评价电影（最高5分）；

Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	Final rating
5	4	5	4	4	4

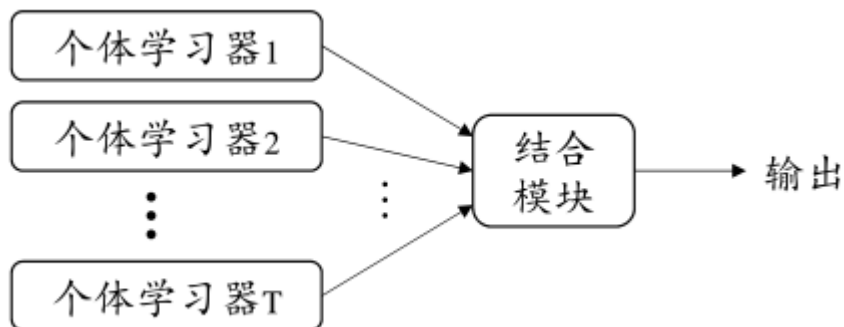
这里对每个数据点的多次预测进行平均。在这种方法中，采用所有模型中取平均值作为最终预测。平均法可用于在回归问题中进行预测或在计算分类问题的概率时使用。

加权平均法

	Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	Final rating
weight	0.23	0.23	0.18	0.18	0.18	
rating	5	4	5	4	4	4.41

集成学习法(Ensemble learning)通过将多个分类学习方法聚集在一起来提高分类准确率。

- 通常一个集成分类器的分类性能会好于单个分类器。集成学习法由训练数据构建一组基分类器(base classifier)，然后通过对每个基分类器的预测进行投票来进行分类。
- 在构建分类器的过程中，一般有两种集成方法：
 - 一种是使用训练集的不同子集训练得到不同的基分类器。
 - 另一种是使用同一个训练集的不同属性子集训练得到不同的基分类器。



高级的集成技术：Bagging

- (1) 从样本集中重采样(有重复的)选出 n 个样本
- (2) 在所有属性上，对这 n 个样本建立分类器(ID3、C4.5、CART、SVM、Logistic回归等)
- (3) 重复以上两步 m 次，即获得了 m 个分类器
- (4) 将数据放在这 m 个分类器上，最后根据这 m 个分类器的投票结果，投票次数最多的类别，指定为最终的输出。

测试例1 测试例2 测试例3				测试例1 测试例2 测试例3				测试例1 测试例2 测试例3			
h_1	√	√	×	h_1	√	√	×	h_1	√	×	×
h_2	×	√	√	h_2	√	√	×	h_2	×	√	×
h_3	√	×	√	h_3	√	√	×	h_3	×	×	√
集群	√	√	√	集群	√	√	×	集群	×	×	×
(a) 集群提升性能				(b) 集群不起作用				(c) 集群起负作用			

√表示分类正确，X表示分类错误。

- Bagging通过降低基分类器的方差改善了泛化误差。
- Bagging的性能依赖于基分类器的稳定性。
 - 如果基分类器是不稳定的（比如：决策树，神经网络算法。），装袋有助于降低训练数据的随机波动导致的误差；
 - 如果基分类器是稳定的（基分类器对训练数据集中的微小变化是鲁棒的），则集成分类器的误差主要是由基分类器的偏倚所引起的，bagging可能不会对基分类器的性能有明显改善。
 - 另外由于每一个样本被选中的概率相同，因此装袋并不侧重于训练数据集中的任何特定实例。因此对于噪声数据，装袋不太受过分拟合的影响。

随机森林引入

随机森林 (Random Forest , 简称RF) 是Bagging的一个扩展变体。RF在以**决策树**为基学习器构建Bagging集成的基础上, 进一步在决策树的训练过程中引入了**随机属性选择**。

传统决策树在选择划分属性时是在当前结点的属性集合(假定有 d 个属性)中选择一个最优属性;

在RF中, 对基决策树的每个结点, 先从该结点的属性集合中随机选择一个包含 k 个属性的子集, 然后再从这个子集中选择一个最优属性用于划分。

这里的参数 k 控制了随机性的引入程度: 若令 $k=d$, 则基决策树的构建与传统决策树相同; 若令 $k=1$, 则是随机选择一个属性用于划分。

随机森林

随机森林是一个树型分类器 $\{h(x, B_k), k=1, \dots\}$ 的集合。其中基分类器 $h(x, B_k)$ 是用CART算法构建的没有剪枝的分类回归树；

随机森林的输出采用简单多数投票法（针对分类）或单颗树输出结果的简单平均（针对回归）得到。

对于随机森林的理解

将若干个弱分类器的分类结果进行投票选择，从而组成一个强分类器，这就是随机森林bagging的思想

关于bagging的一个有必要提及的问题：bagging的代价是不用单棵决策树来做预测，具体哪个变量起到重要作用变得未知，所以bagging改进了预测准确率但损失了解释性。

影响随机森林分类性能的主要因素

- 森林中单颗树的分类强度（Strength）：每颗树的分类强度越大，则随机森林的分类性能越好。
- 森林中树之间的相关度（Correlation）：树之间的相关度越大，则随机森林的分类性能越差。

根据已有训练集生成了对应的随机森林，如何利用某个人的5个特征（年龄、性别、教育、行业、居住地）来预测他的收入层次：

收入层次：

Band 1: Below \$40,000

Band 2: \$40,000 – 150,000

Band 3: More than \$150,000

	Salary Band	1	2	3
Age	Below 18	90%	10%	0%
	19-27	85%	14%	1%
	28-40	70%	23%	7%
	40-55	60%	35%	5%
	More than 55	70%	25%	5%

	Salary Band	1	2	3
Gender	Male	70%	27%	3%
	Female	75%	24%	1%

	Salary Band	1	2	3
Education	<=High School	85%	10%	5%
	Diploma	80%	14%	6%
	Bachelors	77%	23%	0%
	Master	62%	35%	3%

	Salary Band	1	2	3
Residence	Metro	70%	20%	10%
	Non-Metro	65%	20%	15%

	Salary Band	1	2	3
Industry	Finance	65%	30%	5%
	Manufacturing	60%	35%	5%
	Others	75%	20%	5%

我们预测某个人信息：Age：35；Gender：male；Education:Diploma；Industry：manufacturing；Residence:Metro

CART	Band	1	2	3
Age	28-40	70%	23%	7%
Gender	Male	70%	27%	3%
Education	Diploma	80%	14%	6%
Industry	Manufacturing	60%	35%	5%
Residence	Metro	70%	20%	10%
Final probability		70%	24%	6%

最后，我们得出结论，这个人收入层次70%是1等，24%是2等,3%是3等，最终该人属于1等收入层次（小于\$40,000）

随机森林的优点

- 在当前所有算法中，具有极好的准确率；
- 训练可以高度并行化，能够有效地运行在大数据集上；
- 能够处理具有高维特征的输入样本，而且不需要降维；
- 对部分特征缺失不敏感。

随机森林拥有广泛的应用前景，从市场营销到医疗保健保险，既可以用来做市场营销模拟的建模，统计客户来源，保留和流失，也可用来预测疾病的风险和病患者的易感性。