

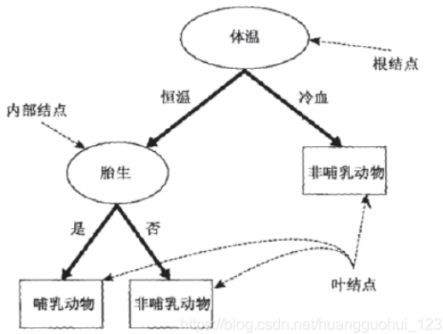
一、决策树原理

决策树（Decision Tree）是一种非参数的有监督学习方法，它能够从一系列有特征和标签的数据中总结出决策规则，并用树状图的结构来呈现这些规则，以解决分类和回归问题。决策树算法容易理解，适用各种数据，在解决各种问题时都有良好表现，尤其是以树模型为核心的各种集成算法，在各个行业和领域都有广泛的应用。

我们来简单了解一下决策树是如何工作的。决策树算法的本质是一种图结构，我们只需要问一系列问题就可以对数据进行分类了。比如说，来看看下面这组数据集，这是一系列已知物种以及所属类别的数据：

名字	体温	表皮覆盖	胎生	水生动物	飞行动物	有腿	冬眠	类标号
人类	恒温	毛发	是	否	否	是	否	哺乳类
鲑鱼	冷血	鳞片	否	是	否	否	否	鱼类
鲸	恒温	毛发	是	否	否	否	否	哺乳类
青蛙	冷血	无	否	半	否	是	是	两栖类
巨蜥	冷血	鳞片	否	否	否	是	否	爬行类
蝙蝠	恒温	毛发	是	否	是	是	否	哺乳类
鸽子	恒温	羽毛	否	否	是	是	否	鸟类
猫	恒温	软毛	是	否	否	是	否	哺乳类
豹纹鲨	冷血	鳞片	是	半	否	是	否	鱼类
海龟	冷血	鳞片	否	半	否	是	否	爬行类
企鹅	恒温	羽毛	否	半	否	是	否	鸟类
豪猪	恒温	刚毛	是	否	否	是	否	哺乳类
鳎	冷血	鳞片	否	是	否	否	是	鱼类
蜥蜴	冷血	无	否	半	否	是	是	两栖类

根据已经收集到的数据，决策树算法为我们算出了下面的这棵决策树：



根据以上的决策树规则，当我们发现一个新的物种，我们就可以根据决策树判断它所属的类别。

**根节点：**没有进边，有出边。包含最初的，针对特征的提问。

**中间节点（内部节点）：**既有进边也有出边，进边只有一条，出边可以有很多条。都是针对特征的提问。

**叶子节点：**有进边，没有出边，每个叶子节点都是一个类别标签。

**子节点和父节点：**在两个相连的节点中，更接近根节点的是父节点，另一个是子节点。

二、构建决策树

任意一个数据集上的所有特征都可以被拿来分枝，每个特征上的任意值都可以当成一个节点自由组合（例如身高可以分为<1.8米，>=1.8米，也可以分为<1米，>=1米，等等），所以一个数据集上可以发展出非常多棵决策树，其数量可达指数级。在这些树中，总有那么一棵树比其他的树分类效果都好，那样的树叫做“全局最优树”。

**全局最优：**经过组合形成的，整体来说分类效果最好的模型

**局部最优：**每一次分枝的时候都向着更好的分类效果分枝，但无法确认如此生成的树在全局上是否是最优的。

**决策树算法的核心是要解决两个问题：**

- 1) 如何从数据表中找出最佳节点和最佳分枝？
- 2) 如何让决策树停止生长，防止过拟合？

一个数据集可以发展的决策树数量是指数级别的，在这么多的决策树当中挑选出一颗“全局最优树”，是非常低效的，于是我们牺牲掉小部分的准确率，用逐步的“局部最优”来达到最接近“全局最优”的结果，这类算法我们统称为“贪心算法”。

**贪心算法：**通过实现局部最优来达到接近全局最优结果的算法，所有的树模型都是这样的算法。

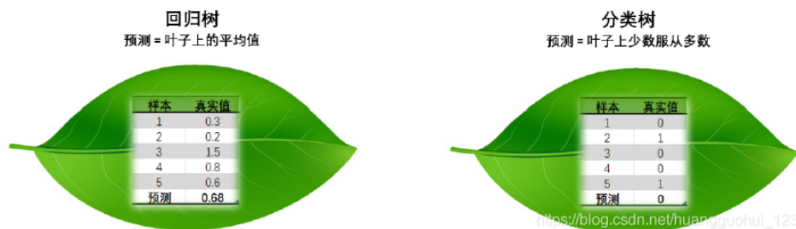
最典型的决策树算法是Hunt算法，该算法是由Hunt等人提出的最早的决策树算法。现代，Hunt算法是许多决策树算法的基础，包括ID3、C4.5和CART等。Hunt算法诞生时间较早，且基础理论并非特别完善，此外以应用较广、理论基础较为完善的ID3算法的基本原理开始，讨论如何利用局部最优优化方法来创建决策模型。

ID3算法原型见于J.R Quinlan的博士论文，是基础理论较为完善，使用较为广泛的决策树模型。在此基础上J.R Quinlan进行优化后，陆续推出了C4.5和C5.0决策树算法，后二者现已称为当前最流行的决策树算法，我们先从ID3开始讲起，再讨论如何从ID3逐渐优化至C4.5。

为了要将数据集转化为一棵树，决策树需要找出最佳节点和最佳的分枝方法，而衡量这个“最佳”的指标叫做“不纯度”。不纯度基于叶子节点来计算的，所以树中的每个节点都会有一个不纯度，并且子节点的不纯度一定是低于父节点的，也就是说，在同一棵决策树上，叶子节点的不纯度一定是最低的。

**不纯度：**决策树的每个叶子节点中都会包含一组数据，在这组数据中，如果有某一类标签占有较大的比例，我们就说叶子节点“纯”，分枝分得好。某一类标签占的比例越大，叶子就越纯，不纯度就越低，分枝就越好。

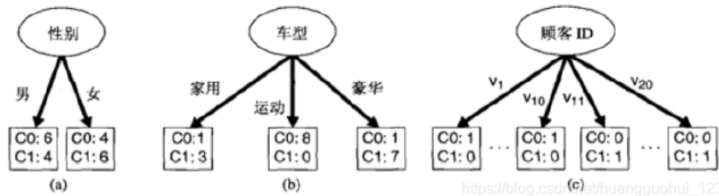
如果没有哪一类标签的比例很大，各类标签都相对平均，则说叶子节点“不纯”，分枝不好，不纯度高。



如上图右边的分类树，**分类型决策树在叶子节点上的决策规则是少数服从多数**，在一个叶子节点上，如果某一类标签所占的比例较大，那所有进入这个叶子节点的样本都会被认为是这一类别（5个样本当中，标签1为2个，标签0为3个，则预测该节点上所有的样本都为标签0）。

如果叶子本身不纯，那测试样本就很有可能被判断错误，相对的叶子越纯，那样本被判断错误的可能性就越小，决策树对训练集的拟合越好。

如下图，（a）的不纯度最高，（c）的不纯度最低。



### 三、量化不纯度

#### 1、误差率 (Classification Error)

不管不纯度的度量方法如何，都是由误差率衍生而来，其计算公式如下：

$$Classification\ error(t) = 1 - \max_{i=1} [p(i|t)]$$

$t$ 代表决策树的某节点， $D_t$ 是 $t$ 节点所对应的数据集，设 $p(i|t)$ 表示给定结点 $t$ 中属于类别的样本 $i$ 所占的比例，这个比例越高，则代表叶子越纯。相应的误差率越低。由此还衍生出了其他两个常用指标，一个是ID3中信息增益 (Information gain) 用到的 **信息熵** (Entropy)，即最为人熟知的信息熵，又叫做香农熵。另一个指标则是基尼指数 (Gini)，主要用于CART决策树的纯度判定。

#### 2、信息熵 (Entropy)

信息熵，又叫做香农熵，其计算公式如下：

$$Entropy(t) = - \sum_{i=0}^{c-1} p(i|t) \log_2 p(i|t)$$

其中 $c$ 表示叶子节点上标签类别的个数， $c-1$ 表示标签的索引。注意在这里，是从第0类标签开始计算，所以最后的标签类别应该是总共 $c$ 个标签， $c-1$ 为最后一个标签的索引。在计算Entropy时设定 $\log_2 0 = 0$ 。

#### 3、基尼指数 (Gini)

$$Gini = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2$$

**案例1：**假设在二分类问题中各节点呈现如下分布，则可进一步计算上述三指数的结果

结点 $N_1$	计数	$Gini = 1 - (0/6)^2 - (6/6)^2 = 0$
类 = 0	0	$Entropy = -(0/6)\log_2(0/6) - (6/6)\log_2(6/6) = 0$
类 = 1	6	$Error = 1 - \max[0/6, 6/6] = 0$

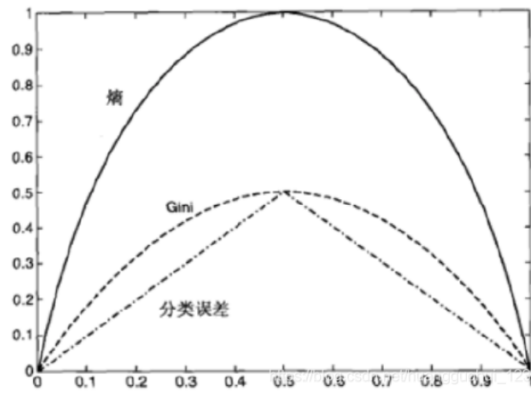
  

结点 $N_2$	计数	$Gini = 1 - (1/6)^2 - (5/6)^2 = 0.278$
类 = 0	1	$Entropy = -(1/6)\log_2(1/6) - (5/6)\log_2(5/6) = 0.650$
类 = 1	5	$Error = 1 - \max[1/6, 5/6] = 0.167$

结点 $N_3$	计数	$Gini = 1 - (3/6)^2 - (3/6)^2 = 0.5$
类 = 0	3	$Entropy = -(3/6)\log_2(3/6) - (3/6)\log_2(3/6) = 1$
类 = 1	3	$Error = 1 - \max[3/6, 3/6] = 0.5$

能够看出，三种方法本质上都相同，在类分布均衡时（即当 $p=0.5$ 时）达到最大值，而当所有记录都属于同一个类时（ $p$ 等于1或0）达到最小值。换言之，在纯度较高时三个指数均较低，而当纯度较低时，三个指数都比较大，且可以计算得出，熵在0-1区间内分布，而Gini指数和分类误差率均在0-0.5区间内分布，三个指数随某变量占比增加而变化的曲线如下图所示：



#### 4. 信息增益 (Information gain)

ID3采用信息熵来衡量不纯度，ID3最优条件是叶节点的总信息熵最小，因此ID3决策树在决定是否对某节点进行切分的时候，会尽可能选取使得该节点对应的子节点信息熵最小的特征进行切分。换言之，就是要求父节点信息熵和子节点总信息熵之差要最大。对于ID3而言，二者之差就是信息增益。

一个父节点下可能有多个子节点，而每个子节点又有自己的信息熵，所以父节点信息熵和子节点信息熵之差，应该是(父节点的信息熵)-(所有子节点信息熵的加权平均)。其中，权重=(单个叶子节点上的样本量)/(父节点上的总样本量)。

父节点和子节点的不纯度下降数可由下述公式进行计算：

$$\Delta info = I(parent) - I(child)$$

其中 $\Delta info$ 代表信息增益， $I(parent)$ 代表父节点的信息熵， $I(child)$ 代表与该父节点相连的所有子节点信息熵的加权平均：

$$I(child) = \sum_{j=1}^k \frac{N(v_j)}{N} I(v_j)$$

其中， $N$ 是父结点上的样本数， $k$ 是这一层上子节点的个数， $N(v_j)$ 是与子结点 $v_j$ 相关联的样本个数， $I(v_j)$ 代表某一子节点的不纯度度量（即基尼系数或者信息熵）。

#### 案例2

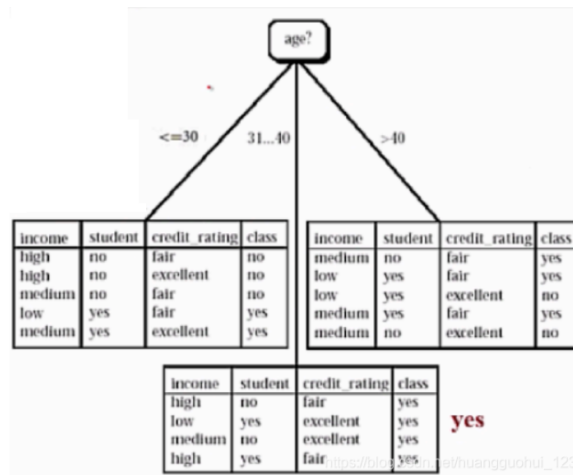
有如下数据集，是一个消费者个人属性和信用评分数据，标签是"是否会发生购买电脑行为"，是一个典型的二分类问题，在此数据集之上我们使用ID3构建决策树模型，并提取有效的分类规则。

	A	B	C	D	E	F
1	ID	age	income	student	credit_rating	Class: buys_computer
2		1 <=30	high	no	fair	no
3		2 <=30	high	no	excellent	no
4		3 31...40	high	no	fair	yes
5		4 >40	medium	no	fair	yes
6		5 >40	low	yes	fair	yes
7		6 >40	low	yes	excellent	no
8		7 31...40	low	yes	excellent	yes
9		8 <=30	medium	no	fair	no
10		9 <=30	low	yes	fair	yes
11		10 >40	medium	yes	fair	yes
12		11 <=30	medium	yes	excellent	yes
13		12 31...40	medium	no	excellent	yes
14		13 31...40	high	yes	fair	yes
15		14 >40	medium	no	excellent	no

(1) 由原始数据集计算总的信息熵 (Entropy) ,即根节点的信息熵，标签为二分类，(class='yes'):(class='no')=9:5。

根据公式  $Entrop(t) = - \sum_{i=0}^{c-1} p(i/t) \log_2 p(i/t)$  , 得  $I(s1, s2) = I(9, 5) = -\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14} = 0.940$

(2) 对原始数据集按照age进行划分



(3) 计算子节点的信息熵的加权平均值

对于  $\text{age} \leq 30$ ,  $(\text{class} = \text{'yes'}):(\text{class} = \text{'no'}) = 2:3$ ,  $I(2, 3) = -\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} = 0.971$

对于  $30 < \text{age} \leq 40$ ,  $(\text{class} = \text{'yes'}):(\text{class} = \text{'no'}) = 4:0$ ,  $I(4, 0) = -\frac{4}{4} \log_2 \frac{4}{4} - \frac{0}{4} \log_2 \frac{0}{4} = 0$

对于  $\text{age} > 40$ ,  $(\text{class} = \text{'yes'}):(\text{class} = \text{'no'}) = 3:2$ ,  $I(3, 2) = -\frac{3}{5} \log_2 \frac{3}{5} - \frac{2}{5} \log_2 \frac{2}{5} = 0.971$

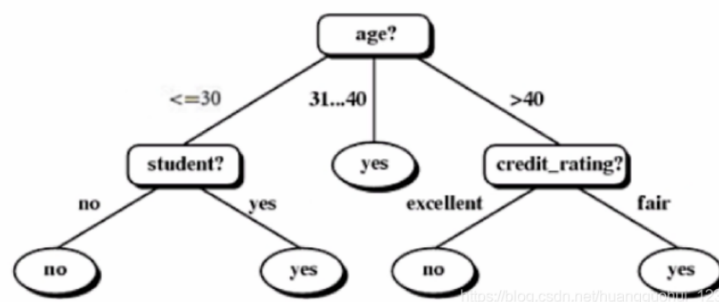
$Entrop(\text{age}) = \frac{5}{14} I(2, 3) + \frac{4}{14} I(4, 0) + \frac{5}{14} I(3, 2) = 0.694$

(4) 计算信息增益

$\Delta \text{info}(\text{age}) = I(9, 5) - Entrop(\text{age}) = 0.940 - 0.694 = 0.246$

(5) 以此类推, 我们还能计算其他几个特征的信息增益, 最终计算结果如下  $\Delta \text{info}(\text{income}) = 0.029$ ,  $\Delta \text{info}(\text{student}) = 0.15$ ,  $\Delta \text{info}(\text{credit\_ating}) = 0.048$ 。很明显, 第一次切分过程将采用age字段进行切分。

(6) 第一次切分完成后, 分成了三个数据集, 其中  $30 < \text{age} \leq 40$  分类指标所指向的数据集纯度为1, 因此不再进行切分, 而其他两个数据集则需要进一步进行切分, 对于  $\text{age} \leq 30$  的数据集而言使用student特征进行切分后子节点纯度就将为1, 而  $\text{age} > 40$  的数据集则可采用credit\_rating字段进行切分。最终ID3决策树模型如下所示:



总的来说, 决策树模型是一个典型的贪心模型, 总目标是一个全局最优解, 即一整套合理的分类规则使得最终叶节点的纯度最高, 但全局最优解在随特征增加而呈现指数级增加的搜索空间内很难高效获取, 因此我们退而求其次, 考虑采用局部最优来一步步推导结果——只要保证信息增益最大, 我们就能得到次最优的模型。当然, 局部最优不一定等于全局最优, 接下来我们就ID3的局限性及优化的方法进行讨论。

#### 四、ID3的局限性

ID3局限主要源于信息增益的计算方法, 其局限性主要有以下几点:

- 分支度越高 (一个父节点分成n个子节点) 的离散变量往往子节点的总信息熵会小。ID3是按照某一列进行切分, 比如有一用户信息表, 极端情况下取用户ID (键) 作为切分字段, 每个分类的纯度都是100%, 因此这样的分类方式是没有效益的。C4.5算法中对此进行了优化。
- 不能直接处理连续型变量, 若需要使用ID3处理连续型变量, 则首先需要连续变量进行离散化, 在C4.5算法中, 同样还增加了针对连续变量的处理手段。
- 对缺失值较为敏感, 使用ID3之前需要提前对缺失值进行处理
- 没有剪枝的设置, 容易导致“过拟合”, 即在训练集上表现很好, 测试集上表现很差

#### 五、C4.5算法

##### 1、修改局部最优优化条件

在C4.5中, 首先通过引入分支度 (Information Value) 的概念, 来对信息增益的计算方法进行修正, 简而言之, 就是在信息增益计算方法的子节点总信息熵的计算方法中添加了随着分类变量水平的惩罚项 (即一个父节点分出来的子节点越多, 惩罚越多)。而分支度的计算公式仍然是基于熵的算法, 只是将信息熵计算公式中的  $p(i/t)$  (即某类别样例占总样例数) 改成了  $p(v_i)$  (即某子节点的总样本数占父节点总样本数的比例), 这其实就是我们加权求和时的“权重”。分支度计算公式如下:

$$\text{Information Value} = - \sum_{i=1}^k P(v_i) \log_2 P(v_i)$$

其中， $k$ 是这一层上子节点的个数， $i$ 表示父节点的第 $i$ 个子节点， $v_i$ 表示第 $i$ 个子节点样例数， $P(v_i)$ 表示第 $i$ 个子节点拥有样例数占父节点总样例数的比例。很明显，IV可作为惩罚项带入子节点的信息熵计算中。

$k$ 的取值范围为[1,父节点的样本例数]，我们考虑两个极端情况，当 $k=1$ 时（即一个父节点仅分出一个子节点），IV值为 $-\left(\frac{k}{k}\right)\log_2\left(\frac{k}{k}\right)=0$ ，IV值为 $-\sum_{i=1}^k\left(\frac{1}{k}\right)\log_2\left(\frac{1}{k}\right)=-\log_2\left(\frac{1}{k}\right)=-\log_2k^{-1}=\log_2k$ 。所以IV值会随着叶子节点上样本量的变小（分支数变大）而逐渐变大，这就是说一个特征中如果标签分类太多，每个叶子上的IV值就会非常大。

这样的分支度指标，让我们在切分的时候，自动避免那些分类水平太多，信息熵减小过快的特征影响模型，减少过拟合情况。

最终，在C4.5中，使用之前的信息增益除以分支度作为选取切分字段的参考指标，该指标被称作Gain Ratio（获利比例，或增益率），计算公式如下：

$$\text{Gain Ratio} = \frac{\text{Information Gain}}{\text{Information Value}}$$

**增益比例是我们决定对哪一列进行分枝的标准**，我们分枝的是GR最大的那一列，本质是信息增益最大，分支度又较小的列（也就是纯度提升很快，但又不是靠着把类别分别细末提升的那些特征）。IV越大，即某一列的分类水平越多，Gain ratio实现的惩罚比例越大。当然，我们还是希望GR越大越好。

我们继续上面的案例2，计算增益率。

(7) 计算信息增益率

首先，分支度  $IV(\text{age}) = -\frac{5}{14}\log_2\frac{5}{14} - \frac{4}{14}\log_2\frac{4}{14} - \frac{5}{14}\log_2\frac{5}{14} = 1.577$

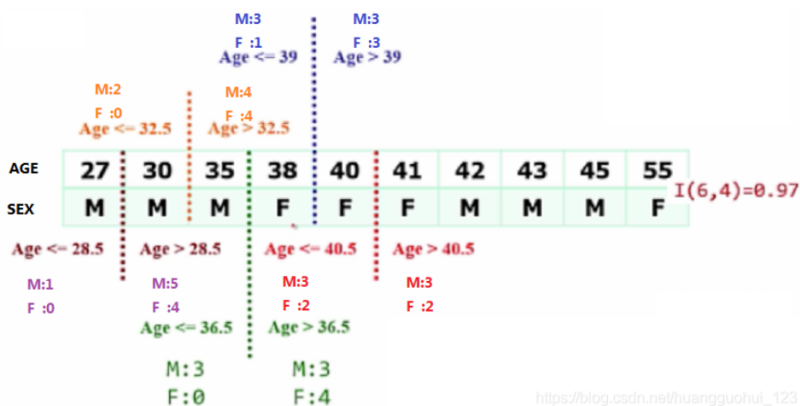
则信息增益率  $GR(\text{age}) = \frac{\Delta\text{info}(\text{age})}{IV(\text{age})} = \frac{0.246}{1.577} = 0.156$

然后可进一步计算其他各字段的GR值，并选取GR值最大的字段进行切分。

## 2、连续变量处理手段

在C4.5中，同样还增加了针对连续变量的处理手段。如果输入特征字段是连续型变量，则有下列步骤：

1. 算法首先会对这一列数进行从小到大的排序
  2. 选取相邻的两个数的中间数作为切分数据集的备选点，若一个连续变量有N个值，则在C4.5的处理过程中将产生N-1个备选切分点，并且每个切分点都代表着一种二叉树的切分方案。
- 这里需要注意的是，此时针对连续变量的处理并非是其转化为一个拥有N-1个分类水平的分类变量，而是将其转化成了N-1个二分方案。例如有如下数据集，数据集中只有两个字段，第一行代表年龄，是特征变量，第二行代表性别，是目标字段，则对年龄这一连续变量的切分方案如图所示：



找信息增益率（Gain Ratio）最大的切点

全部数据集的信息熵： $I(6, 4) = 0.97$

1) 以age=36.5为切点：

$I(3, 0) = 0$ ,  $I(3, 4) = 0.99$

$Entropy = 0.3 * 0 + 0.7 * 0.99 = 0.69$

$Information\ Gain = 0.97 - 0.69 = 0.28$

$Information\ Value = I(3, 7) = 0.88$

$Gain\ Ratio = 0.28 / 0.88 = 0.32$

2) 以age=39为切点：

$I(3, 1) = 0.81$ ,  $I(3, 3) = 1$

$Entropy = 0.4 * 0.81 + 0.6 * 1 = 0.924$

$Information\ Gain = 0.97 - 0.924 = 0.046$

$Information\ Value = I(4, 6) = 0.97$

$Gain\ Ratio = 0.46 / 0.97 = 0.47$

...

以此类推，算出每一个切分方案的信息增益率（gain ratio），找出最佳切分方案。

从上述论述能够看出，在对于包含连续变量的数据集进行树模型构建的过程中要消耗更多的运算资源。但与此同时，我们也会发现，当连续变量的某中间点参与到决策树的二分过程中，往往代表该点对于最终分类结果有较大影响，这也为我们连续变量的分箱压缩提供了指导性意见，例如上述案例，若要对age列进行压缩，则可考虑使用36.5对其进行分箱，则分箱结果对于性别这一目标字段仍然具有较好的分类效果，这也是决策树的最常见用途之一，也是最重要的模型指导分箱的方法。

## 六、CART算法

现在被大量使用的是从C4.5改进的CART树，CART树本质其实和C4.5区别不大，**只不过CART树所有的层都是二叉树**，也就是每层只有两个分枝。用CART树来走一遍C4.5的流程。假设年龄是特征，性别是标签：

1. 首先将年龄从小到大依此进行排列
2. 然后计算俩俩相邻的年龄的均值
3. 按均值所在的点，对连续性变量进行二分（变成数字形式的，第一类是>均值，第二类是<均值），二分得到的点叫做决策树的“树桩”。这是说，对于一个含有n个记录的连续性变量来说，就会存在n-1个潜在切分点，这n-1个潜在切分点的获益比例都会被计算
4. 找每种二分切分方案的获益比例，获益比例最大的切分点，就是切点
5. 切完之后，计算加权信息熵，计算信息增益，引入分支度，可以计算出增益比例了

**需要注意的是**，这种切分方法，每次切分之后，并没有消费掉一列（特征），只消费掉了一个备选点（即下一次切分，该特征还会有可能继续被切分）。在我们原本的切分方法中，**ID3每次切分就会消费掉一整个列**，但实际上，我们可以只关注每次分类的时候，对信息熵的减少贡献最多的那个分类。按案例2来说，我们在分类age的时候，最关注的是31~40岁的那一个分类（信息熵为0，不纯度最低），我们完全可以实现，31~40一类，其他算一类，然后我们再将“其他”这个类别进行相似的二分类（切分为<30和>40两类）。这就是CART的核心原理，大量地减少了计算的量。

**对于KNN算法**，就是每一个特征对于推断的重要性是一样的。这也是KNN最大的缺陷。而决策树天生就认为每个特征对于推断的重要性是不一样的，而CART数则是进一步认为，每个特征下的每个分类对于推断的重要性也是一样的（是由于其“二分类”的特性导致的）。