1.决策树概念及结构

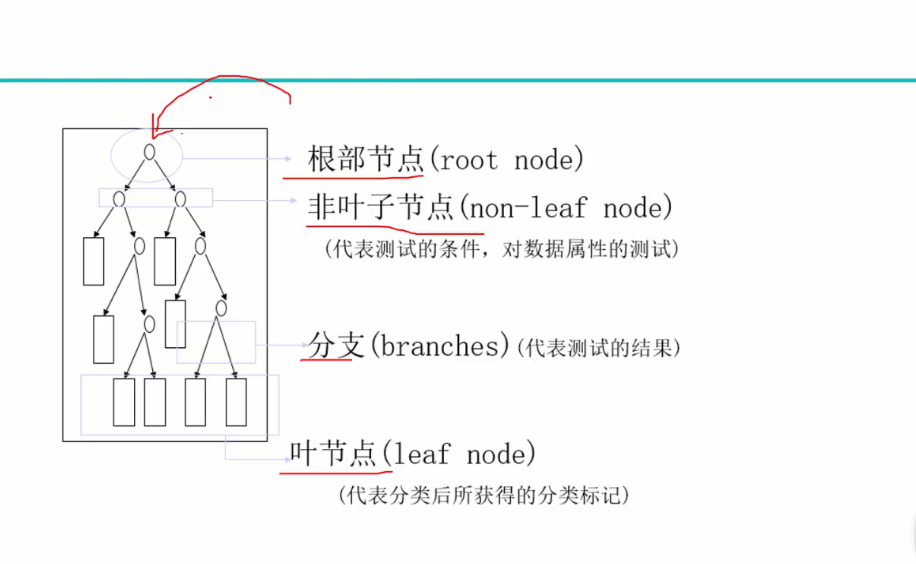
决策树算法是以树状结构表示数据分类的结果，每一个决策点实现一个具有离散输出的测试，如图所示

首先决策树里边有一个根节点，就是数据放决策树当中，它是先走的那一层决策， 。我们说，它是一个根节点。

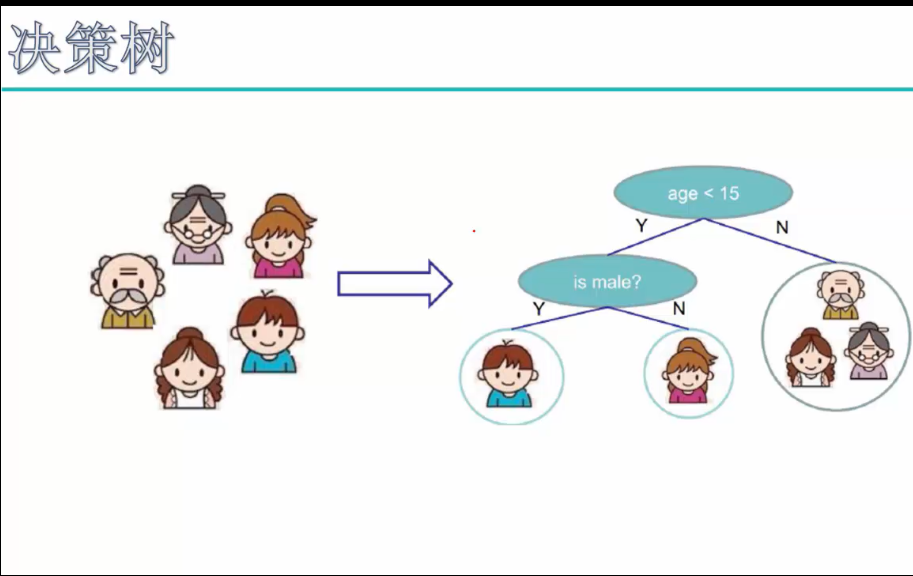
还有什么非叶子节点飞叶子节点的意思！就是说对于这个节点，他下面还是有一个分支！还是有个分之说明这个。节点还没有结束，把它没有得出来一个最终的结果，直我们把这样的节点叫做一个非叶子节点。

那么，什么叫做叶子节点？这样最后我们看最后一个节点。最后这些是不是他就是最后一个，所以说，我们说最后的这样底下不再进行一个分裂了，我们就把这些点叫做一个叶子节点。其实，这些叶子节点里边包含的就是最终想要的一个结果值了。把每一个叶子结点里边都有一个唯一的类别是有这样一个回归，只可以计算出来，

分支就是这样一个杈了，这样一个杈了这样一个杈，那么在这个决策树当中，现在来说一下有这样一个根节点也有非叶子叶子节点还有分支，还有叶节点。



举一个例子来说，假如我要在如图以下五个人中找出一个会打篮球的人。我们发现，这五个人分别是爷爷·奶奶·妈妈·女儿·儿子，我们假定年轻而且是男性的人会打篮球。我可以通过他们的年级是否大于15岁，找出第一组符合年龄条件的人，在通过是男生还是女生选出第二组符合性别条件的人，于是，我们通过两次决策来选出所需要的选择。



2.熵及信息增益

当然我们也可以通过先选择是男性在选择小于15岁的挑选出需要的人从而做出决策。到底哪种方式好呢，这时候我们就可以引入熵这个概念

A熵

熵通俗的讲是就是混乱程度，这里我们称不确定性程度。

但是通过调查，我们可以把不确定变为确定，我们把消除不确定的东西成为信息。

从定义**来讲：** 当一件事情有多种可能情况时，这件事情对某人而言具体是哪种情况的不确定性叫做**熵**，而能够消除该人对这件事情不确定性的事物叫做**信息**。

**熵和信息的关系：**数量相等，意义相反。获取信息 = 消除熵。

对应的我们把不能消除不确定的东西叫做噪声，通常来说我们进行决策依赖的数据，既包含信息，也包含噪声。这时候我们就要对不确定性程度进行量化，以得知数据的准确程度

我们可以这样想，

我们投1个硬币，他有2种情况

我们投2个硬币，他有4种情况

我们投3个硬币，他有8种情况

我们投m个硬币，他有种情况

……

我们也可以这样说，

* 定义：抛1次硬币的不确定性为1Bit，出现结果为2种等概率事件(这里的1Bit就是单位信息量)。
* 那么，抛2次硬币的不确定性为2Bit,出现结果为4种等概率事件，即（00/01/10/11）
* 那么，抛3次硬币的不确定性为3Bit,出现结果为8种等概率事件（000/001/010/011/100/101/110/111）。
* 那么，抛N次硬币的不确定性为NBit，出现结果为2^N=M种等概率事件，这里 因此，假设我们知道出现了16种等概率事件(即16种可能情况）,含有的信息量为

以上注意：1）信息是指消除事物的不确定性;2）单位信息是指抛一枚硬币所产生的不确定性。在现实生活中，如何理解信息是指事件的不确定性？

但是我们发现投硬币投出正反两种情况概率是相等的，在现实中，往往，某种情况出现的概率，与另一种情况出现的概率是不等的

比如转盘

这时候，我们可以假设

三等奖是由三个事件组合而成

二等奖是由两个事件组合而成

一等奖是由一个事件组合而成

这些事件概率相等

总熵就是ent=6

那么三等奖他的熵ent1=log6-=-log（3/6）

同理二等奖 ent2=log6-log2=-log（2/6）

一等奖 ent3=log6-log1=-log(1/6)

而获得每一中奖都是概率不一样的，总熵为ent总=-【（3/6）ent1+（2/6）ent2+（1/6）ent3】

即Ent=…………………………………………………………………………………….注1

其中p表示这个事件出现的概率

拿以下例子举例

**例1：**当小明不会某道数学选择题时（正确答案是C）。

正确答案（宏观态）是 A，B，C，D 哪个选项（4 个微观态）的不确定性就是熵。

此时熵ent=-log(1/4)=2（bit）

**宏观态与微观态：**这里，正确答案也叫宏观态，而每个可能选项叫做微观态。宏观态是不考虑内部细节的状态，而微观态是考虑具体细节的状态。

**熵与信息的大小：** 熵在 A，B，C，D 所有可能情况（宏观态）都是等概率（1/4）时最大，在确定 C 了（实际情况）是 100% 后最小。

**信息的种类：**能够消除不确定性的信息有三种类型，它们本质都是正确的调整每个可能情况（微观态）的概率。

**第一种：**能正确的调整某件事情的可能情况（微观态）的概率。

**第二种：**能正确的排除某件事情的干扰情况。

**例1：**小红告诉小明 “D 选项是错的”。

此时熵

这句话帮助小明将 D 选项的概率调整到了 0%（0.415 bits），这时小明只需要从 3 种情况里确定实际情况即可。

**例2：**在此基础上，再告诉小明 “A选项是错的”。

这句话帮助小明将 A 选项的概率调整到了 0%（0.585 bits），这时小明只需要从 2 种情况里确定实际情况即可。

**例3：**在此基础上，再告诉小明 “B选项是错的”。

这句话帮助小明将 B 选项的概率调整到了 0%（1 bits），只剩下一种情况了，不确定性（熵）完全消除。

**第三种：**能够直接确定某件事情的实际情况。

**例1：**小红告诉小明 “正确答案是 C”。

是将 C 选项的概率调整到了100%（2 bits），这句话帮助小明从 4 种等概率情况里确定了实际情况。

**非信息：**未能消除不确定性不叫信息。

**例1：**但小红告诉小明 “肯定是ABCD里的一项”（0 bit）。

这句话就没有帮助小明消除任何不确定性，信息为 0 。

在这种定义下，并没有假信息一说。 因为只有能够消除某人对某件事情的不确定性的事物才是信息，因此小红告诉小明“正确答案是 D”提供的信息是 0。

那些不能够消除某人对某件事情不确定性的事物被称为数据或噪音。

噪音是干扰某人获得信息的事物。而数据是噪音与信息的混合，需要用知识将其分离。

2决策树算法

a信息增益与ID3算法

现在回答刚开始提出的问题：（当然我们也可以通过先选择是男性在选择小于15岁的挑选出需要的人从而做出决策。到底哪种方式好呢）

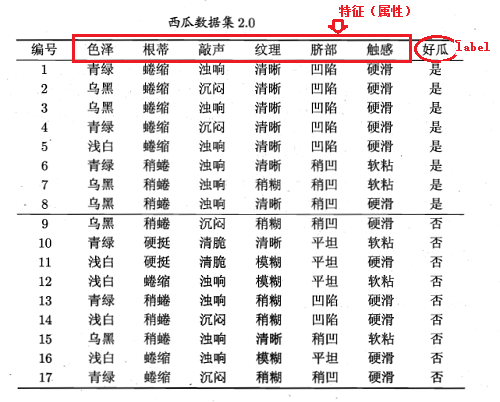
需要明确，构造决策树的基本想法是随着树的深度增加 节点的熵迅速降低，熵降低的速度越快越好，我们有望得到一颗最矮的决策树

此时熵最小，那么让原来的熵减去现在的熵就能表示上减少了多少，这个就是

信息增益gain=ent原-ent后，

当然，我们挑选信息增益大的决策树，这种算法有称为是ID3算法

**举例子用的数据集为：**

****

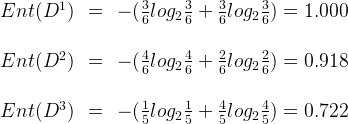
**显然该数据集包含17个样本，类别为二元的，即。则，正例（类别为1的样本）占的比例为：，反例（类别为0的样本）占的比例为：。根据信息熵的公式能够计算出数据集D的信息熵为：**



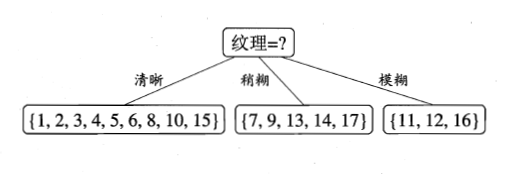
**从数据集中能够看出特征集为：{色泽、根蒂、敲声、纹理、脐部、触感}。下面我们来计算每个特征的信息增益。先看“色泽”，它有三个可能的离值：{青绿、乌黑、浅白}，若使用“色泽”对数据集D进行划分，则可得到3个子集，分别为：** **（色泽=青绿）、** **（色泽=乌黑）、** **（色泽=浅白）。** **共包含6个样本{1，4，6，10，13，17}** **，其中正例占** **，反例占** **。**

**** **包含6个样本{2，3，7，8，9，15}** **，其中正例占** **，反例**

**占。包含了5个样本{5，11，12，14，16}，其中正例占，反例占。因此，可以计算出用“色泽”划分之后所获得的3个分支结点的信息熵为：**

****

**比较发现，特征“纹理”的信息增益最大，于是它被选为划分属性。因此可得：**

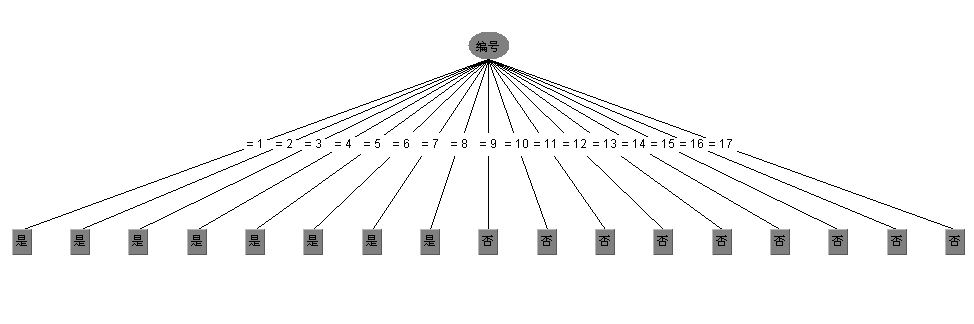
****

此后每一步与第一步相同

b.信息增益率 CT4算法

信息增益是有局限性的

**但信息增益有个缺点就是对可取数值多的属性有偏好，举个例子讲，还是考虑西瓜数据集，如果我们把“编号”这一列当做属性也考虑在内，那么可以计算出它的信息增益为0.998，远远大于其他的候选属性，因为“编号”有17个可取的数值，产生17个分支，每个分支结点仅包含一个样本，显然这些分支结点的纯度最大。但是，这样的决策树不具有任何泛化能力。还是拿西瓜数据集2.0来测试下，如果考虑编号这一属性，看看ID3算法会生成一颗什么样的决策树：**

****

显然是生成了一颗含有17个结点的树，他的熵为零，增益系数最大。但是这棵树只对这17个训练用的样本有分辨能力，对其他样本例如18，或者测试样本没有分辨能力。也就是说没没有任何的泛化能力，这也是信息增益的一个缺点

由于信息增益的缺点，提出了对其的改进信息增益率来构造决策树，此外他为了避免过拟合，还提供了剪枝的功能。他能够处理连续值与缺失值。来看看信息增益率：



当信息增益率最大时，我们选择该种决策树也就是CT4.5算法

C.基尼系数

基尼系数和信息增益率类似



不过多赘述

D.C5.0算法C5.0是C4.5应用于大数据集上的分类算法，主要在执行效率和内存使用方面进行了改进。

3.剪枝

A预剪枝

预剪枝就是在构建决策树的过程提前停止进一步细分，准确的说，预剪枝是要对划分前后泛化性能进行评估。对比决策树某节点生成前与生成后的泛化性能。

还拿西瓜例子来说

我们可以从样例数，列数等来限制决策树的深度

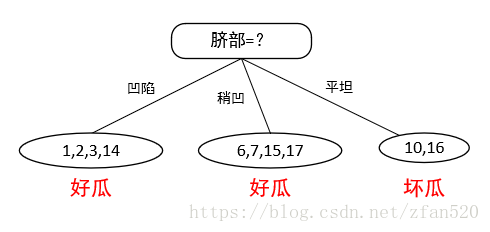
拿样本数举例，当所分的样本中个数少于一定的值，且精度足够后不再进行划分，下面是精细讲解

（1）在未划分前，根据训练集，类别标记为训练样例数最多的类别，由于训练集中的好瓜与坏瓜是相同多的类别，均为5，因此任选其中一类，书中选择了好瓜作为标记类别。

当所有节点集中在根节点，所有训练集属于标记类别的仅有{4,5,8}，因此分类正确的是3/7\*100%=42.9%

|  |  |
| --- | --- |
| 编号 | 好瓜(正确结果) |
| 4 | 是 |
| 5 | 是 |
| 8 | 是 |
| 9 | 否 |
| 11 | 否 |
| 12 | 否 |
| 13 | 否 |
|  | 3/7 |

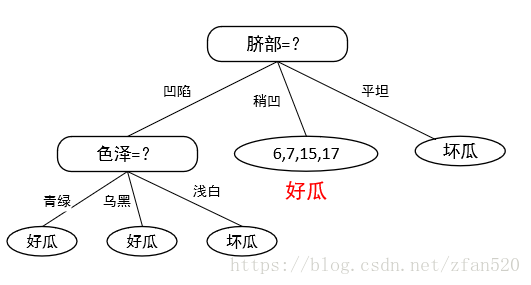
（2）计算训练集的信息增益，得知脐部的信息增益最大，因此按照脐部进行划分。又因为在训练集中，凹陷特征好瓜的占比多，因此凹陷划分为好瓜，稍凹特征好过占比多，因此将其标记为好瓜，因此按照脐部划分的子树结果如下：



划分后，对比结果如下：

|  |  |  |
| --- | --- | --- |
| 编号 | 好瓜(正确结果) | 按照脐部划分 |
| 4(凹陷) | 是 | 是 |
| 5(凹陷) | 是 | 是 |
| 8(稍凹) | 是 | 是 |
| 9(稍凹) | 否 | 是(划分错误) |
| 11(平坦) | 否 | 否 |
| 12(平坦) | 否 | 否 |
| 13(凹陷) | 否 | 是(划分错误) |
| 正确率 | 3/7 | 5/7(精度提高，划分) |

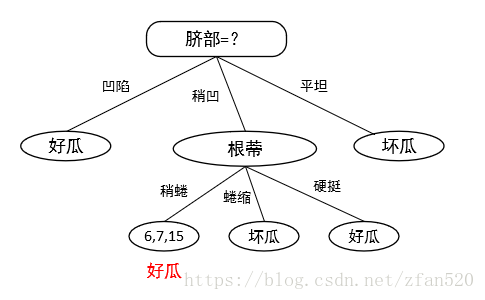
（3）在脐部划分的基础上，进一步计算凹陷、根蒂特征下，其他属性的信息增益，根据计算结果可知，在凹陷的情况下，色泽的信息增益最大，因此对于凹陷的西瓜，进一步确定按照色泽进行划分，划分结果如下：



对于凹陷数据，进一步按照色泽进行划分后，对比划分前后的准确性：

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 好瓜(正确结果) | 按照脐部划分 | 对凹陷，按照色泽划分 |
| 4(凹陷、青绿) | 是 | 是 | 是 |
| 5(凹陷、浅白) | 是 | 是 | 否 |
| 8(稍凹) | 是 | 是 | 是(不满足条件的，按照上一次划分结果照写) |
| 9(稍凹) | 否 | 是(划分错误) | 是 |
| 11(平坦) | 否 | 否 | 否 |
| 12(平坦) | 否 | 否 | 否 |
| 13(凹陷、青绿) | 否 | 是(划分错误) | 是 |
| 正确率 | 3/7 | 5/7(精度提高，划分) | 4/7(精度降低，不划分) |

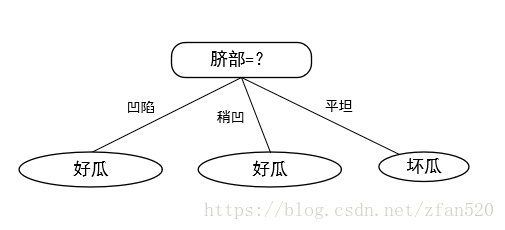
对稍凹数据集，进一步计算其他属性的信息增益，确定根蒂的信息增益最大，因此对稍凹，进一步按照根蒂进行划分：



对于稍凹数据，进一步按照根蒂进行划分后，对比划分前后的准确性：

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 好瓜(正确结果) | 按照脐部划分 | 对稍凹，按照根蒂划分 |
| 4(凹陷) | 是 | 是 | 是（不满足条件的，按照上次换发你结果照写） |
| 5(凹陷) | 是 | 是 | 是 |
| 8(稍凹、蜷缩) | 是 | 是 | 否 |
| 9(稍凹、蜷缩) | 否 | 是(划分错误) | 否 |
| 11(平坦) | 否 | 否 | 否 |
| 12(平坦) | 否 | 否 | 否 |
| 13(凹陷) | 否 | 是(划分错误) | 是 |
| 正确率 | 3/7 | 5/7(精度提高，划分) | 5/7(精度不变，不划分) |

（4）因此按照预剪枝，最终形成的决策树如下图，其泛化性为71.4%。



由图可知，预剪枝使得很多分支没有展开，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。但是，有些分支虽当前不能提升泛化性。甚至可能导致泛化性暂时降低，但在其基础上进行后续划分却有可能导致显著提高，因此预剪枝的这种贪心本质，给决策树带来了欠拟合的风险。

后剪枝

后剪枝就是在整个决策树建立好之后，通过裁减减少决策树深度

可以通过CaT=C(T)+aT计算

其中C(T)=·H(t) Nt代表叶子结点的样本数，H（t）代表基尼系数

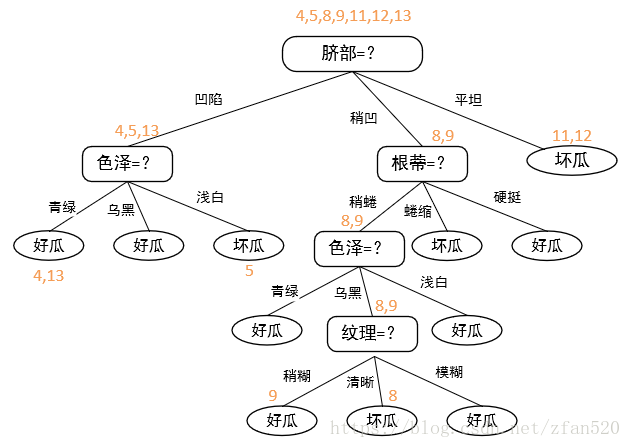
T代表叶子数

具体事例如下

**3、后剪枝**

后剪枝表示先从训练集中生成一颗完整决策树。

（1）我在此生成的决策树上将测试集的数据在此树上进行了标记，如下图所示：

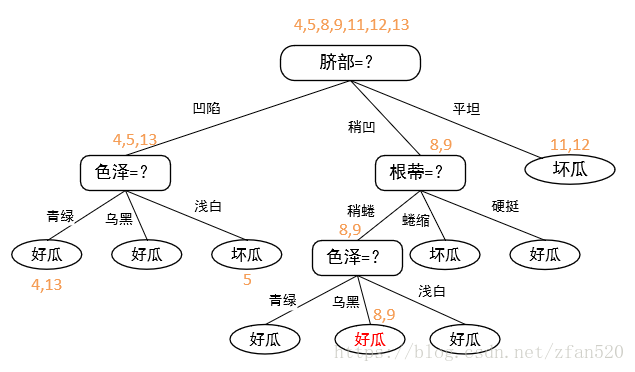


对比标记节点的划分类与各数据的真实分类，计算准确率，如下表所示：

|  |  |  |
| --- | --- | --- |
| 编号 | 好瓜(正确结果) | 按照整棵树进行划分 |
| 4 | 是 | 是 |
| 5 | 是 | 否 |
| 8 | 是 | 否 |
| 9 | 否 | 是 |
| 11 | 否 | 否 |
| 12 | 否 | 否 |
| 13 | 否 | 是 |
| 正确率 |  | 3/7 |

生成的决策树，在验证集上的准确度为3/7\*100%=42.9%

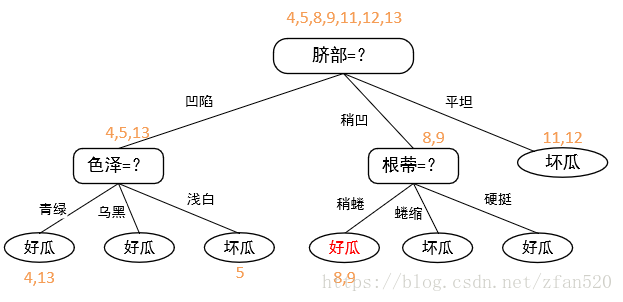
（2）后剪枝将从决策树的底部往上进行剪枝，先看最底部的纹理，将其领衔的分支减掉，即将其换成叶子节点。由于在训练集上，替换后，包含的样本号为{7,15}，好瓜坏瓜比例相等，因此选择好瓜进行标记，剪枝后的决策树为：



|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 好瓜(正确结果) | 按照整棵树进行划分 | 减掉底部纹理划分 |
| 4 | 是 | 是 | 是(其他不变，照写) |
| 5 | 是 | 否 | 否 |
| 8 | 是 | 否 | 是(改变) |
| 9 | 否 | 是 | 是 |
| 11 | 否 | 否 | 否 |
| 12 | 否 | 否 | 否 |
| 13 | 否 | 是 | 是 |
| 正确率 |  | 3/7 | 4/7(准确率提高) |

当减掉底部纹理划分后，准确率提高，因此按照纹理划分需裁剪掉。

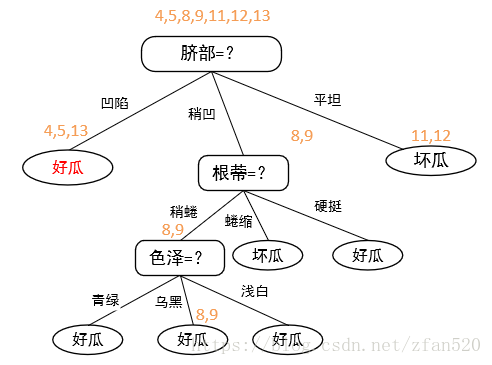
（3）接着往上裁剪，此时应该是色泽部分，由于在训练集上，替换后，包含的样本号为{6,7,15}，好瓜（2个）多于坏瓜（1个），因此选择好瓜进行标记，剪枝后的决策树为：



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 好瓜(正确结果) | 按照整棵树进行划分 | 减掉底部纹理划分 | 减掉底部色泽划分 |
| 4 | 是 | 是 | 是(其他不变，照写) | 是(其他不变，照写) |
| 5 | 是 | 否 | 否 | 否 |
| 8 | 是 | 否 | 是(改变) | 是 |
| 9 | 否 | 是 | 是 | 是 |
| 11 | 否 | 否 | 否 | 否 |
| 12 | 否 | 否 | 否 | 否 |
| 13 | 否 | 是 | 是 | 是 |
| 正确率 |  | 3/7 | 4/7(准确率提高) | 4/7(准确率不变) |

此时决策树验证集精度仍为57.1%，因此可不进行剪枝，即对于脐部稍凹，根蒂稍蜷部分，可保留按照色泽进一步划分。

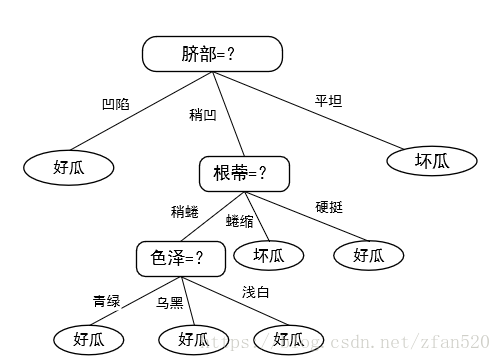
（4）接下来，我们看脐部凹陷分支。由于在训练集上，将色泽替换为叶节点后，包含的样本号为{1,2,3,14}，好瓜（3个）多于坏瓜（1个），因此选择好瓜进行标记，剪枝后的决策树为：



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 好瓜(正确结果) | 按照整棵树进行划分 | 减掉底部纹理划分 | 减掉底部色泽划分 | 减调色泽划分(最左侧色泽) |
| 4 | 是 | 是 | 是(其他不变，照写) | 是(其他不变，照写) | 是 |
| 5 | 是 | 否 | 否 | 否 | 是(新划分，发生改变) |
| 8 | 是 | 否 | 是(改变) | 是 | 是(其他不变，照写) |
| 9 | 否 | 是 | 是 | 是 | 是 |
| 11 | 否 | 否 | 否 | 否 | 否 |
| 12 | 否 | 否 | 否 | 否 | 否 |
| 13 | 否 | 是 | 是 | 是 | 是 |
| 正确率 |  | 3/7 | 4/7(准确率提高) | 4/7(准确率不变) | 5/7(准确率提高) |

当减掉最左侧色泽划分后，准确率提高，因此按照色泽划分需裁剪掉。

（5）整棵树遍历基本完成，因此该决策树最终后剪枝的结果如下图所示，其验证精度为71.4%。



**4、总结**

对比预剪枝与后剪枝生成的决策树，可以看出，后剪枝通常比预剪枝保留更多的分支，其欠拟合风险很小，因此后剪枝的泛化性能往往由于预剪枝决策树。但后剪枝过程是从底往上裁剪，因此其训练时间开销比前剪枝要大。