[**决策树系列（五）——CART**](http://www.cnblogs.com/yonghao/p/5135386.html)

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，又能是分类树；

（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。

接下来将以一个实际的例子对CART进行介绍：

                                                        　　　　　　　　　　　　表1 原始数据表

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

**从以下的思路理解CART：**

**分类树？回归树？**

      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。

      CART既能是分类树，又能是决策树，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：

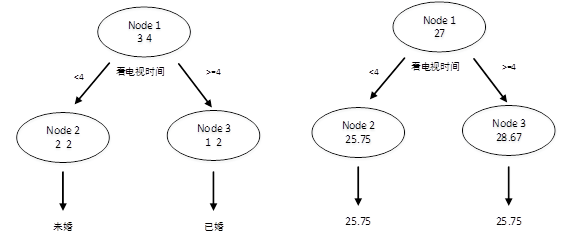


                        　　　　　　　　　　　　　　图1 预测婚姻情况决策树                                               图2 预测年龄的决策树

       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；

       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。

**CART如何选择分裂的属性？**

      分裂的目的是为了能够让数据变纯，使决策树输出的结果更接近真实值。那么CART是如何评价节点的纯度呢？如果是分类树，CART采用GINI值衡量节点纯度；如果是回归树，采用样本方差衡量节点纯度。节点越不纯，节点分类或者预测的效果就越差。

GINI值的计算公式：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140130616-1716397508.png

      节点越不纯，GINI值越大。以二分类为例，如果节点的所有数据只有一个类别，则https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140147319-2077011507.png ，如果两类数量相同，则https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140156163-1303524251.png 。

回归方差计算公式：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140203741-208915813.png

      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。

      因此，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）：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140229210-576757285.png

或者（回归树）：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140250444-1000189848.png

**CART如何分裂成一棵二叉树？**

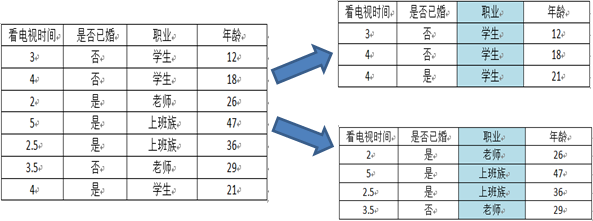
     节点的分裂分为两种情况，连续型的数据和离散型的数据。

     CART对连续型属性的处理与C4.5差不多，通过最小化分裂后的GINI值或者样本方差寻找最优分割点，将节点一分为二，在这里不再叙述，详细请看**C4.5**。

     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，怎么办呢？很简单，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，举一个简单的例子：如果某离散属性一个有三个离散值X，Y，Z，则该属性的分裂方法有{X}、{Y，Z}，{Y}、{X，Z}，{Z}、{X，Y}，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。

     以属性“职业”为例，一共有三个离散值，“学生”、“老师”、“上班族”。该属性有三种划分的方案，分别为{“学生”}、{“老师”、“上班族”}，{“老师”}、{“学生”、“上班族”}，{“上班族”}、{“学生”、“老师”}，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，如下图所示：

**第一种划分方法：{“学生”}、{“老师”、“上班族”}**



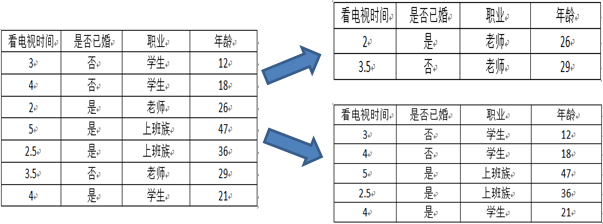
预测是否已婚（分类）：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140514319-212672665.png

预测年龄（回归）：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140542975-1587608963.png

**第二种划分方法：{“老师”}、{“学生”、“上班族”}**



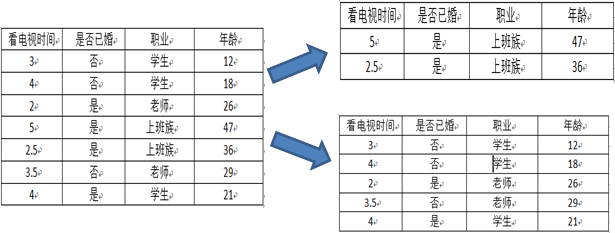
预测是否已婚（分类）：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140635835-1539341444.png

预测年龄（回归）：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140642710-1920641216.png

**第三种划分方法：{“上班族”}、{“学生”、“老师”}**



 预测是否已婚（分类）：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140714210-336892119.png

预测年龄（回归）：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140721725-571821361.png

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。

**如何剪枝？**

      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相同小，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。

可描述如下：

令决策树的非叶子节点为https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140800835-1653693022.png。

a）计算所有非叶子节点的表面误差率增益值https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140810803-1471070338.png

b）选择表面误差率增益值https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140819507-171604546.png最小的非叶子节点https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140831866-73536628.png（若多个非叶子节点具有相同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https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140844600-1126502165.png进行剪枝

表面误差率增益值的计算公式：

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140855991-920761949.png

其中：

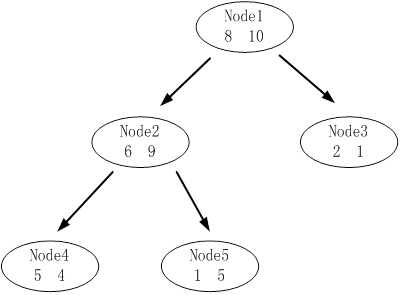
https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140910569-63573161.png表示叶子节点的误差代价，https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140918553-2097300878.png ，https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140925116-1852988504.png 为节点的错误率， https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140932991-878466782.png为节点数据量的占比；

https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140944397-1411086671.png表示子树的误差代价，https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140951022-1776394064.png ， https://images2015.cnblogs.com/blog/833682/201601/833682-20160116140957491-164197365.png为子节点i的错误率，https://images2015.cnblogs.com/blog/833682/201601/833682-20160116141003007-1453455827.png 表示节点i的数据节点占比；

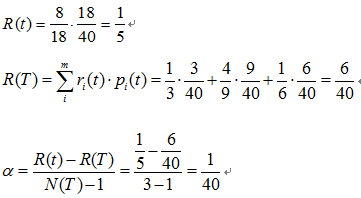
https://images2015.cnblogs.com/blog/833682/201601/833682-20160116141013882-1398176582.png表示子树节点个数。

算例：

下图是其中一颗子树，设决策树的总数据量为40。



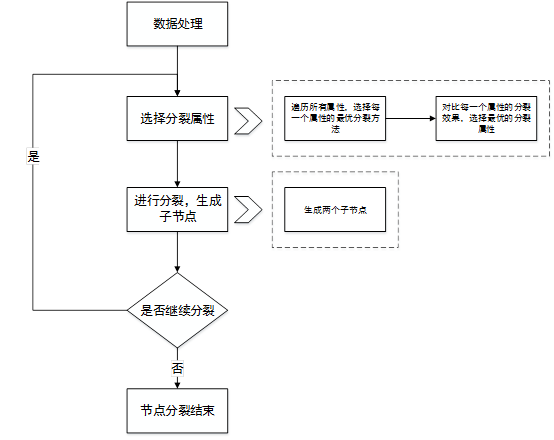
该子树的表面误差率增益值可以计算如下：



求出该子树的表面错误覆盖率为 ，只要求出其他子树的表面误差率增益值就可以对决策树进行剪枝。

**程序实际以及源代码**

**流程图：**

****

（1）数据处理

         对原始的数据进行数字化处理，并以二维数据的形式存储，每一行表示一条记录，前n-1列表示属性，最后一列表示分类的标签。

         如表1的数据可以转化为表2：

                                                                           表2 初始化后的数据

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

      其中，对于“婚姻情况”属性，数字{1，2}分别表示{未婚，已婚 }；对于“职业”属性{1，2，3， }分别表示{学生、老师、上班族}；

代码如下所示：

         static double[][] allData;                              //存储进行训练的数据

　　　 static List<String>[] featureValues;                    //离散属性对应的离散值

featureValues是链表数组，数组的长度为属性的个数，数组的每个元素为该属性的离散值链表。

（2）两个类：节点类和分裂信息

a）节点类Node

      该类表示一个节点，属性包括节点选择的分裂属性、节点的输出类、孩子节点、深度等。注意，与ID3中相比，新增了两个属性：leafWrong和leafNode\_Count分别表示叶子节点的总分类误差和叶子节点的个数，主要是为了方便剪枝。

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif 树的节点

b）分裂信息类，该类存储节点进行分裂的信息，包括各个子节点的行坐标、子节点各个类的数目、该节点分裂的属性、属性的类型等。

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif 分裂信息

主方法findBestSplit(Node node,List<int> nums,int[] isUsed)，该方法对节点进行分裂

其中：

node表示即将进行分裂的节点；

nums表示节点数据对一个的行坐标列表；

isUsed表示到该节点位置所有属性的使用情况；

findBestSplit的这个方法主要有以下几个组成部分：

1）节点分裂停止的判定

节点分裂条件如上文所述，源代码如下：

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif 停止分裂的条件

2）寻找最优的分裂属性

寻找最优的分裂属性需要计算每一个分裂属性分裂后的GINI值或者样本方差，计算公式上文已给出，其中GINI值的计算代码如下：

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif GINI值计算

3）进行分裂，同时对子节点进行迭代处理

其实就是递归的过程，对每一个子节点执行findBestSplit方法进行分裂。

findBestSplit源代码：

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif 节点选择属性和分裂

（4）剪枝

代价复杂度剪枝方法（CCP）：

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif CCP代价复杂度剪枝

CART全部核心代码：

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif CART核心代码

**总结：**

（1）CART是一棵二叉树，每一次分裂会产生两个子节点，对于连续性的数据，直接采用与C4.5相似的处理方法，对于离散型数据，选择最优的两种离散值组合方法。

（2）CART既能是分类数，又能是二叉树。如果是分类树，将选择能够最小化分裂后节点GINI值的分裂属性；如果是回归树，选择能够最小化两个节点样本方差的分裂属性。

（3）CART跟C4.5一样，需要进行剪枝，采用CCP（代价复杂度的剪枝方法）