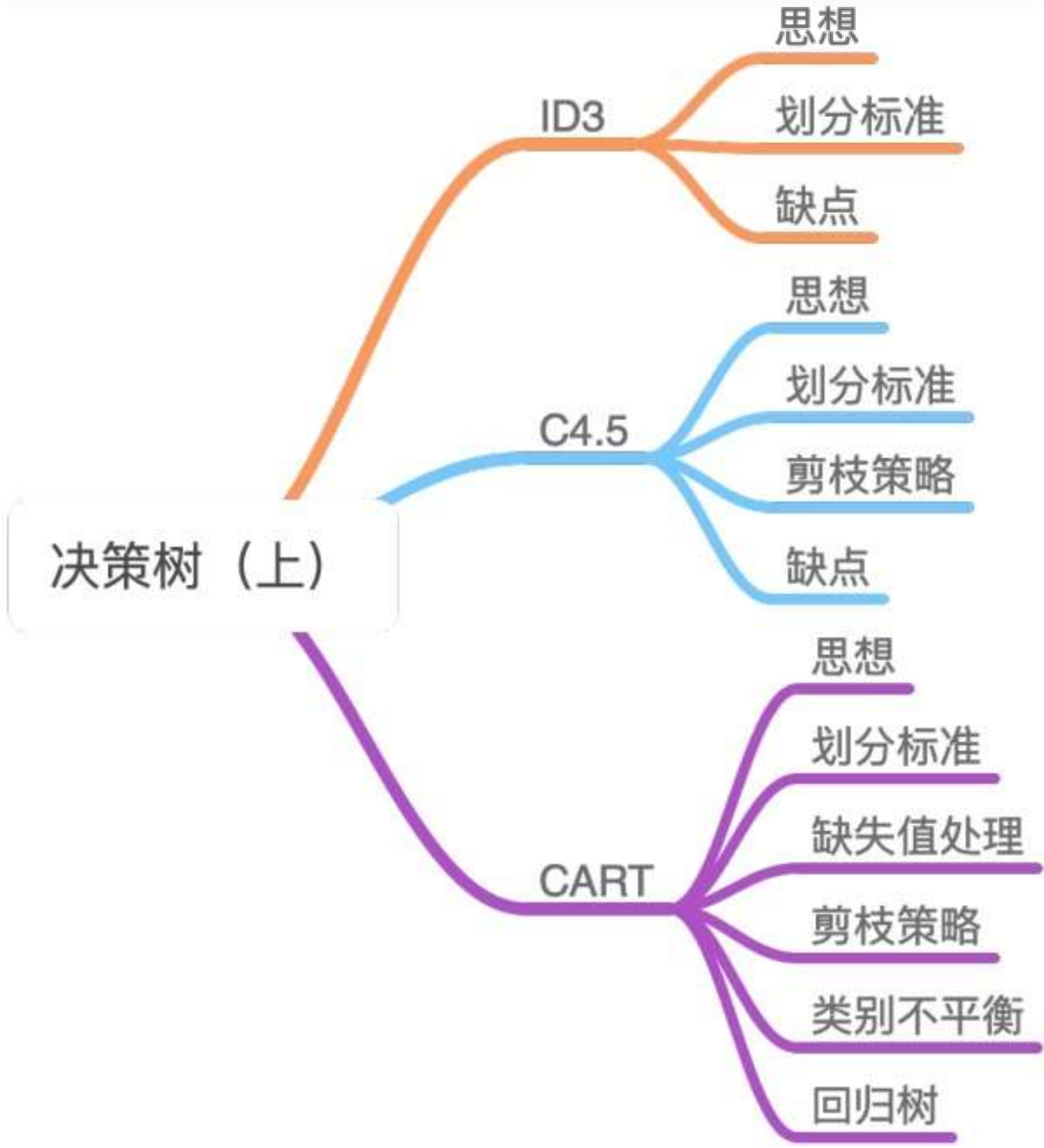

【机器学习】决策树（上）——ID3、C4.5、CART（非常详细）

决策树是一个非常常见并且优秀的机器学习算法，它易于理解、可解释性强，其可作为分类算法，也可用于回归模型。本文将分三篇介绍决策树，第一篇介绍基本树（包括 ID3、C4.5、CART），第二篇介绍 Random Forest、Adaboost、GBDT，第三篇介绍 Xgboost 和 LightGBM。

对于基本树我将大致从以下四个方面介绍每一个算法：思想、划分标准、剪枝策略，优缺点。

1. ID3

ID3 算法是建立在奥卡姆剃刀（用较少的东西，同样可以做好事情）的基础上：越是小型的决策树越优于大的决策树。

1.1 思想

从信息论的知识中我们知道：信息熵越大，从而样本纯度越低，。ID3 算法的核心思想就是以信息增益来度量特征选择，选择信息增益最大的特征进行分裂。算法采用自顶向下的贪婪搜索遍历可能的决策树空间（C4.5 也是贪婪搜索）。 其大致步骤为：

1. 初始化特征集合和数据集合；
2. 计算数据集合信息熵和所有特征的条件熵，选择信息增益最大的特征作为当前决策节点；
3. 更新数据集合和特征集合（删除上一步使用的特征，并按照特征值来划分不同分支的数据集合）；
4. 重复 2，3 两步，若子集值包含单一特征，则为分支叶子节点。

1.2 划分标准

ID3 使用的分类标准是信息增益，它表示得知特征 A 的信息而使得样本集合不确定性减少的程度。

数据集的信息熵：

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

其中 C_k 表示集合 D 中属于第 k 类样本的样本子集。

针对某个特征 A，对于数据集 D 的条件熵 $H(D|A)$ 为：

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

其中 D_i 表示 D 中特征 A 取第 i 个值的样本子集， D_{ik} 表示 D_i 中属于第 k 类的样本子集。

信息增益 = 信息熵 - 条件熵：

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

信息增益越大表示使用特征 A 来划分所获得的“纯度提升越大”。

- ID3 没有剪枝策略，容易过拟合；
- 信息增益准则对可取值数目较多的特征有所偏好，类似“编号”的特征其信息增益接近于 1；
- 只能用于处理离散分布的特征；
- 没有考虑缺失值。

2. C4.5

C4.5 算法最大的特点是克服了 ID3 对特征数目的偏重这一缺点，引入信息增益率来作为分类标准。

2.1 思想

C4.5 相对于 ID3 的缺点对应有以下改进方式：

- 引入悲观剪枝策略进行后剪枝；
- 引入信息增益率作为划分标准；
- 将连续特征离散化，假设 n 个样本的连续特征 A 有 m 个取值，C4.5 将其排序并取相邻两样本值的平均数共 m-1 个划分点，分别计算以该划分点作为二元分类点时的信息增益，并选择信息增益最大的点作为该连续特征的二元离散分类点；
- 对于缺失值的处理可以分为两个子问题：
- 问题一：在特征值缺失的情况下进行划分特征的选择？（即如何计算特征的信息增益率）
- 问题二：选定该划分特征，对于缺失该特征值的样本如何处理？（即到底把这个样本划分到哪个结点里）
- 针对问题一，C4.5 的做法是：对于具有缺失值特征，用没有缺失的样本子集所占比重来折算；
- 针对问题二，C4.5 的做法是：将样本同时划分到所有子节点，不过要调整样本的权重值，其实也就是以不同概率划分到不同节点中。

2.2 划分标准

利用信息增益率可以克服信息增益的缺点，其公式为

$$Gain_{ratio}(D,A)=\frac{Gain(D,A)}{H_A(D)}$$

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

$H_A(D)$ 称为特征 A 的固有值。

这里需要注意，信息增益率对可取值较少的特征有所偏好（分母越小，整体越大），因此 C4.5 并不是直接用增益率最大的特征进行划分，而是使用一个**启发式方法**：先从候选划分特征中找到信息增益高于平均值的特征，再从中选择增益率最高的。

2.3 剪枝策略

为什么要剪枝：过拟合的树在泛化能力的表现非常差。

2.3.1 预剪枝

在节点划分前来确定是否继续增长，及早停止增长的主要方法有：

- 节点内数据样本低于某一阈值；
- 所有节点特征都已分裂；
- 节点划分前准确率比划分后准确率高。

预剪枝不仅可以降低过拟合的风险而且还可以减少训练时间，但另一方面它是基于“贪心”策略，会带来欠拟合风险。

2.3.2 后剪枝

在已经生成的决策树上进行剪枝，从而得到简化版的剪枝决策树。

C4.5 采用的**悲观剪枝方法**，用递归的方式从低往上针对每一个非叶子节点，评估用一个最佳叶子节点去代替这课子树是否有益。如果剪枝后与剪枝前相比其错误率是保持或者下降，则这棵子树就可以被替换掉。C4.5 通过训练数据集上的错误分类数量来估算未知样本上的错误率。

后剪枝决策树的欠拟合风险很小，泛化性能往往优于预剪枝决策树。但同时其训练时间会大的多。

2.4 缺点

- 剪枝策略可以再优化；
- C4.5 用的是多叉树，用二叉树效率更高；
- C4.5 只能用于分类；
- C4.5 使用的熵模型拥有大量耗时的对数运算，连续值还有排序运算；
- C4.5 在构造树的过程中，对数值属性值需要按照其大小进行排序，从中选择一个分割点，所以只适合于能够驻留于内存的数据集，当训练集大得无法在内存容纳时，程序无法运行。

3. CART

ID3 和 C4.5 虽然在对训练样本集的学习中可以尽可能多地挖掘信息，但是其生成的决策树分支、规模都比较大，CART 算法的二分法可以简化决策树的规模，提高生成决策树的效率。

3.1 思想

CART 包含的基本过程有分裂，剪枝和树选择。

- **分裂：**分裂过程是一个二叉递归划分过程，其输入和预测特征既可以是连续型的也可以是离散型的，CART 没有停止准则，会一直生长下去；
- **剪枝：**采用**代价复杂度剪枝**，从最大树开始，每次选择训练数据熵对整体性能贡献最小的那个分裂节点作为下一个剪枝对象，直到只剩下根节点。CART 会产生一系列嵌套的剪枝树，需要从中选出一颗最优的决策树；
- **树选择：**用单独的测试集评估每棵剪枝树的预测性能（也可以用交叉验证）。

CART 在 C4.5 的基础上进行了很多提升。

- C4.5 为多叉树，运算速度慢，CART 为二叉树，运算速度快；
- C4.5 只能分类，CART 既可以分类也可以回归；
- CART 使用 Gini 系数作为变量的不纯度量，减少了大量的对数运算；
- CART 采用代理测试来估计缺失值，而 C4.5 以不同概率划分到不同节点中；
- CART 采用“基于代价复杂度剪枝”方法进行剪枝，而 C4.5 采用悲观剪枝方法。

3.2 划分标准

熵模型拥有大量耗时的对数运算，基尼指数在简化模型的同时还保留了熵模型的优点。基尼指数代表了模型的不纯度，基尼系数越小，不纯度越低，特征越好。这和信息增益（率）正好相反。

$$
\begin{aligned}
Gini(D) &= \sum_{k=1}^{K} \frac{|C_k|}{|D|}(1-\frac{|C_k|}{|D|}) \\
&= 1-\sum_{k=1}^{K}(\frac{|C_k|}{|D|})^2 \\
Gini(D|A) &= \sum_{i=1}^{n} \frac{|D_i|}{|D|} Gini(D_i)
\end{aligned}
$$

其中 k 代表类别。

基尼指数反映了从**数据集中随机抽取两个样本，其类别标记不一致的概率。**因此基尼指数越小，则数据集纯度越高。基尼指数偏向于特征值较多的特征，类似信息增益。基尼指数可以用来度量任何不均匀分布，是介于 0~1 之间的数，0 是完全相等，1 是完全不相等，

此外，当 CART 为二分类，其表达式为：

$$
Gini(D|A) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2)
$$

我们可以看到在平方运算和二分类的情况下，其运算更加简单。当然其性能也与熵模型非常接近。

那么问题来了：基尼指数与熵模型性能接近，但到底与熵模型的差距有多大呢？

我们知道 $ln(x) = -1 + x + o(x)$ ，所以

$$
\begin{aligned}
H(X) &= -\sum_{k=1}^{K} p_k \ln p_k \\
&\approx \sum_{k=1}^{K} p_k \left(1 - p_k\right)
\end{aligned}
$$

我们可以看到，基尼指数可以理解为熵模型的一阶泰勒展开。这边在放上一张很经典的图：

3.3 缺失值处理

上文说到，模型对于缺失值的处理会分为两个子问题：

1. 如何在特征值缺失的情况下进行划分特征的选择？
2. 选定该划分特征，模型对于缺失该特征值的样本该进行怎样处理？

对于问题 1，CART 一开始严格要求分裂特征评估时只能使用在该特征上没有缺失值的那部分数据，在后续版本中，CART 算法使用了一种惩罚机制来抑制提升值，从而反映出缺失值的影响（例如，如果一个特征在节点的 20% 的记录是缺失的，那么这个特征就会减少 20% 或者其他数值）。

对于问题 2，CART 算法的机制是为树的每个节点都找到代理分裂器，无论在训练数据上得到的树是否有缺失值都会这样做。在代理分裂器中，特征的分值必须超过默认规则的性能才有资格作为代理（即代理就是代替缺失值特征作为划分特征的特征），当 CART 树中遇到缺失值时，这个实例划分到左边还是右边是决定于其排名最高的代理，如果这个代理的值也缺失了，那么就使用排名第二的代理，以此类推，如果所有代理值都缺失，那么默认规则就是把样本划分到较大的那个子节点。代理分裂器可以确保无缺失训练数据上得到的树可以用来处理包含确实值的新数据。

3.4 剪枝策略

采用一种“基于代价复杂度的剪枝”方法进行后剪枝，这种方法会生成一系列树，每个树都是通过将前面的树的某个或某些子树替换成一个叶节点而得到的，这一系列树中的最后一棵树仅含一个用来预测类别的叶节点。然后用一种成本复杂度的度量准则来判断哪棵子树应该被一个预测类别值的叶节点所代替。这种方法需要使用一个单独的测试数据集来评估所有的树，根据它们在测试数据集熵的分类性能选出最佳的树。

我们来看具体看一下代价复杂度剪枝算法：

首先我们将最大树称为 T_0 ，我们希望减少树的大小来防止过拟合，但又担心去掉节点后预测误差会增大，所以我们定义了一个损失函数来达到这两个变量之间的平衡。损失函数定义如下：

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

T 为任意子树， $C(T)$ 为预测误差， $|T|$ 为子树 T 的叶子节点个数， α 是参数， $C(T)$ 衡量训练数据的拟合程度， $|T|$ 衡量树的复杂度， α 权衡拟合程度与树的复杂度。

那么如何找到合适的 α 来使得复杂度和拟合度达到最好的平衡点呢，最好的办法就是另 α 从 0 取到正无穷，对于每一个固定的 α ，我们都可以找到使得 $C_\alpha(T)$ 最小的最优子树 $T(\alpha)$ 。当 α 很小的时候， T_0 是最优子树；当 α 最大时，单独的根节点是这样的最优子树。随着 α 增大，我们可以得到一个这样的子树序列： $T_0, T_1, T_2, T_3, \ldots, T_n$ ，这里的子树 T_{i+1} 生成是根据前一个子树 T_i 剪掉某一个内部节点生成的。

Breiman 证明：将 α 从小增大， $0 = \alpha_0 < \alpha_0 < \ldots < \alpha_n < \infty$ ，在每个区间 $[\alpha_i, \alpha_{i+1})$ 中，子树 T_i 是这个区间里最优的。

这是代价复杂度剪枝的核心思想。

我们每次剪枝都是针对某个非叶节点，其他节点不变，所以我们只需要计算该节点剪枝前和剪枝后的损失函数即可。

对于任意内部节点 t，剪枝前的状态，有 $|T_t|$ 个叶子节点，预测误差是 $C(T_t)$ ；剪枝后的状态：只有本身一个叶子节点，预测误差是 $C(t)$ 。

因此剪枝前以 t 节点为根节点的子树的损失函数是：

$$C_\alpha(T) = C(T_t) + \alpha|T|$$

剪枝后的损失函数是

$$C_\alpha(t) = C(t) + \alpha$$

通过 Breiman 证明我们知道一定存在一个 α 使得 $C_\alpha(T) = C_\alpha(t)$ ，使得这个值为：

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

α 的意义在于，$[\alpha_i, \alpha_{i+1})$ 中，子树 T_i 是这个区间里最优的。当 α 大于这个值是，一定有 $C_\alpha(T) > C_\alpha(t)$ ，也就是剪掉这个节点后都比不剪掉要更优。所以每个最优子树对应的是一个区间，在这个区间内都是最优的。

然后我们对 T_i 中的每个内部节点 t 都计算：

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

$g(t)$ 表示阈值，故我们每次都会减去最小的 T_t 。

3.5 类别不平衡

CART 的一大优势在于：无论训练数据集有多失衡，它都可以将其子冻消除不需要建模人员采取其他操作。

CART 使用了一种先验机制，其作用相当于对类别进行加权。这种先验机制嵌入于 CART 算法判断分裂优劣的运算里，在 CART 默认的分类模式中，总是要计算每个节点关于根节点的类别频率的比值，这就相当于对数据自动重加权，对类别进行均衡。

对于一个二分类问题，节点 node 被分成类别 1 当且仅当：

$$\frac{N_1(node)}{N_1(root)} > \frac{N_0(node)}{N_0(root)}$$

比如二分类，根节点属于 1 类和 0 类的分别有 20 和 80 个。在子节点上有 30 个样本，其中属于 1 类和 0 类的分别是 10 和 20 个。如果 10/20>20/80，该节点就属于 1 类。

通过这种计算方式就无需管理数据真实的类别分布。假设有 K 个目标类别，就可以确保根节点中每个类别的概率都是 1/K。这种默认的模式被称为“先验相等”。

先验设置和加权不同之处在于先验不影响每个节点中的各类别样本的数量或者份额。先验影响的是每个节点的类别赋值和树生长过程中分裂的选择。

3.6 回归树

CART（Classification and Regression Tree，分类回归树），从名字就可以看出其不仅可以用于分类，也可以应用于回归。其回归树的建立算法上与分类树部分相似，这里简单介绍下不同之处。

3.6.1 连续值处理

对于连续值的处理，CART 分类树采用基尼系数的大小来度量特征的各个划分点。在回归模型中，我们使用常见的和方差度量方式，对于任意划分特征 A，对应的任意划分点 s 两边划分成的数据集 D_1 和 D_2 ，求出使 D_1 和 D_2 各自集合的均方差最小，同时 D_1 和 D_2 的均方差之和最小所对应的特征和特征值划分点。表达式为：

$$\min_{a,s}\left[\min_{c_1}\sum_{x_i \in D_1}(y_i - c_1)^2 + \min_{c_2}\sum_{x_i \in D_2}(y_i - c_2)^2\right]$$

其中， c_1 为 D_1 数据集的样本输出均值， c_2 为 D_2 数据集的样本输出均值。

3.6.2 预测方式

对于决策树建立后做预测的方式，上面讲到了 CART 分类树采用叶子节点里概率最大的类别作为当前节点的预测类别。而回归树输出不是类别，它采用的是用最终叶子的均值或者中位数来预测输

4. 总结

最后通过总结的方式对比下 ID3、C4.5 和 CART 三者之间的差异。

除了之前列出来的划分标准、剪枝策略、连续值确实值处理方式等之外，我再介绍一些其他差异：

- **划分标准的差异：**ID3 使用信息增益偏向特征值多的特征，C4.5 使用信息增益率克服信息增益的缺点，偏向于特征值小的特征，CART 使用基尼指数克服 C4.5 需要求 log 的巨大计算量，偏向于特征值较多的特征。
- **使用场景的差异：**ID3 和 C4.5 都只能用于分类问题，CART 可以用于分类和回归问题；ID3 和 C4.5 是多叉树，速度较慢，CART 是二叉树，计算速度很快；
- **样本数据的差异：**ID3 只能处理离散数据且缺失值敏感，C4.5 和 CART 可以处理连续性数据且有多种方式处理缺失值；从样本量考虑的话，小样本建议 C4.5、大样本建议 CART。C4.5 处理过程中需对数据集进行多次扫描排序，处理成本耗时较高，而 CART 本身是一种大样本的统计方法，小样本处理下泛化误差较大；
- **样本特征的差异：**ID3 和 C4.5 层级之间只使用一次特征，CART 可多次重复使用特征；
- **剪枝策略的差异：**ID3 没有剪枝策略，C4.5 是通过悲观剪枝策略来修正树的准确性，而 CART 是通过代价复杂度剪枝。

参考

1. 《机器学习》周志华
2. 《数据挖掘十大算法》吴信东
3. CART 树怎么进行剪枝?

编辑于 2020-11-03 19:06

机器学习　决策树　监督学习

文章被以下专栏收录

机器学习算法与自然语言处理
公号[机器学习算法与自然语言处理] 微信号yizhennotes

机器学习理论与数据竞赛实战
公众号【Coggle数据科学】专注算法竞赛实战分享

机器学习爱好者
公众号:机器学习初学者,知识星球:92416895

cs
computer science

算法工程师面试汇总
智力题、数据结构、机器学习、深度学习、面经等

推荐阅读