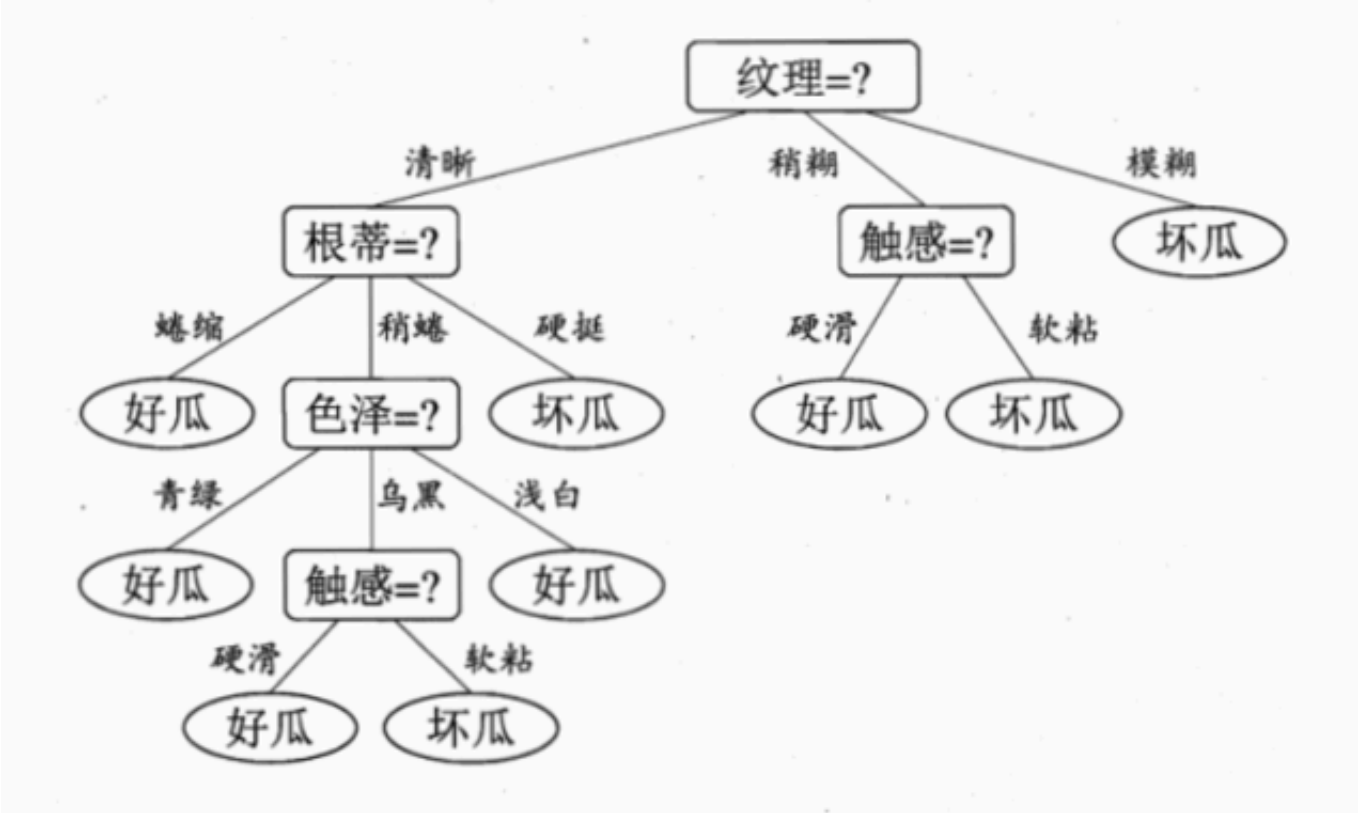
机器学习算法系列（4）：决策树

本文结合李航博士的《统计学习方法》与周志华老师的《机器学习》决策树部分，《统计学习方法》重理论的证明推导，《机器学习》注重讲解算法的特点与扩展。

INTRODUCTION

决策树（Decision Tree）是数据挖掘中一种基本的分类和回归方法，它呈树形结构，在分类问题中，表示基于特征对实例进行分类的过程，可以认为是$if-then$规则的集合，也可认为是定义在特征空间与类空间上的条件概率分布。下图是一个简单的决策树示例：



决策树模型的主要优点是模型具有可读性，分类速度快。在学习时，利用训练数据，根据损失函数最小化原则建立决策树模型；而在预测时，对新的数据，利用决策树模型进行分类。主要的决策树算法有ID3算法、C4.5算法和CART算法。

一个性能良好的决策树，是一个与训练数据矛盾较小的决策树，同时又具有很好地泛化能力。言外之意就是说，好的决策树不仅对训练样本有很好的分类效果，对于测试集也有较低的误差率。一个决策树的学习过程包括三个步骤：特征选择、决策树的生成以及决策树的修剪。

一、决策树模型的两种解释

1.1 决策树模型

分类决策树模型是一种描述对实例进行分类的树形结构。决策树由结点和有向边组成。结点有两种类型：内部结点和叶节点。内部结点表示一个特征或属性，叶节点表示一个类。

1.1.1 决策树与if-then规则

可以将决策树看成一个if-then规则的集合。即由决策树的根结点到叶节点的每一条路径构建一条规则；路径上内部结点的特征对应着规则的条件，而叶结点的类对应着规则的结论。

决策树的路径或其对应的if-then规则集合的重要性质：互斥且完备（每一个实例都被一条路径或一条规则所覆盖，且只被一条路径或一条规则所覆盖，这里的覆盖是指实例的特征与路径上的特征一致或实例满足规则的条件）

1.1.1 决策树与条件概率分布

决策树还表示给定特征条件下类的条件概率分布，它定义在特征空间的一个划分。将特征空间划分为互不相交的单元，并在每个单元定义一个类的概率分布就构成了一个条件概率分布。决策树的每一条路径对应于划分中的一个单元。

假设X为表示特征的随机变量，Y为表示类的随机变量，那么这个条件概率分布可以表示为$P(X|Y)$,各叶结点上的条件概率往往偏向于某一个类，即属于某一类的概率越大。决策树分类时将该结点的实例强行分到条件概率大的那一类去。

二、特征选择

2.1 特征选择问题

若利用一个特征进行分类的结果与随机分类的结果没有很大差异，则称这个特征是没有分类能力的。特征选择的准则是信息增益或信息增益比。直观上，若一个特征具有更好的分类能力，或者说，按照这一特征将训练数据集分割为子集，使得各个子集在当前条件下有最好的分类，那么就更应该选择这个特征。信息增益可以表示这一直观的准则。

2.2 信息增益

2.2.1 熵

在信息论与概率统计中，熵表示随机变量**不确定性**的度量。设X是一个取有限个值得离散随机变量，其概率分布为

$$P\left(X=x_i\right)=p_i, i=1,2,\cdots,n$$

则随机变量X的熵定义为

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

若p_i等于0，定义$0log0 = 0$，熵的单位为比特或者纳特。

2.2.2 条件熵

$H(Y|X)$表示在已知随机变量X的条件下随机变量Y的不确定性定义为X给定条件下Y的条件概率分布的熵对X的数学期望

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

经验熵和经验条件熵：当熵和条件熵中的概率由数据估计（特别是极大似然估计）得到时，所对应的熵与条件熵分别称为经验熵和条件经验熵。

2.2.3 信息增益

信息增益表示得知特征X的信息而使得类Y的信息的不确定性减少的程度。特征A对训练数据集D的信息增益$g(D, A)$，定义为集合D的经验熵$H(D)$与特征A给定条件下D的经验条件熵$H(D|A)$之差，即

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

一般地，熵$H(Y)$与条件熵$H(Y|X)$之差称为互信息。决策树学习中的信息增益等价于训练数据集中类与特征的互信息。

于是我们可以应用信息增益准则来选择特征，信息增益表示由于特征A而使得对数据集D的分类的不确定性减少的程度。对数据集D而言，信息增益依赖于特征，不同的特征往往具有不同的信息增益。信息增益大的特征具有更强的分类能力。

2.2.4 信息增益算法

根据信息增益准则的特征选择方法为对训练数据集（或子集）D，计算其每个特征的信息增益，并比较它们的大小，选择信息增益最大的特征。

在描述算法前，先对符号进行说明：
设训练数据集为D，$|D|$表示其样本容量，即样本个数。设有K个类$C_k, k = 1, 2, \cdots, K, |C_k|$为属于类$C_k$的样本个数，$\sum_{k=1}^{K} |C_k| = |D|$。设特征$A$有$n$个不同的取值$a_1, a_2, \cdots, a_n$,根据特征$A$的取值将$D$划分为$n$个子集$D_1, D_2, \cdots, D_n, |D_i|$为$D_i$的样本个数，$\sum_{i=1}^{n} |D_i| = |D|$。记子集$D_i$

中属于类C_k的样本的集合为D_{ik},即$D_{ik} = D_i \cap C_k$,D_{ik}为D_{ik}的样本个数。

具体算法步骤如下：

- 1）计算数据集D的经验熵$H(D)$

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

- 2）计算特征A对数据集D的经验条件熵$H(D|A)$

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

- 3）计算信息增益

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

2.3 信息增益比

以信息增益作为划分训练数据集的特征，存在偏向于选择取值较多的特征的问题。使用信息增益比可以对这一问题进行校正。

信息增益比表示特征A对训练数据集D的信息增益比。$g_R(D, A)$定义为其信息增益$g(D, A)$与训练数据集D关于特征A的值的熵$H_A(D)$之比，即

$$g_R(D, A) = \frac{g(D, A)}{H_A(D)}$$

2.4 基尼系数

分类问题中，假设有K个类，样本点属于第k类的概率为p_k，则概率分布的基尼系数定义为

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

若样本集合D根据特征A是否取某一可能值a被分割成D_1和D_2两部分，即

$$D_1 = \{(x, y) \in D | A(x) = 0\} \ , \quad D_2 = D - D_1$$

则在特征A的条件下，集合D的基尼指数定义为

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

基尼系数Gini(D)表示集合D的不确定性，表示经A=a分割后集合D的不确定性。基尼系数越大，样本集合的不确定性越大，与熵类似。

从下图可以看出基尼指数和熵之半的曲线很接近，都可以近似地代表分类误差率。

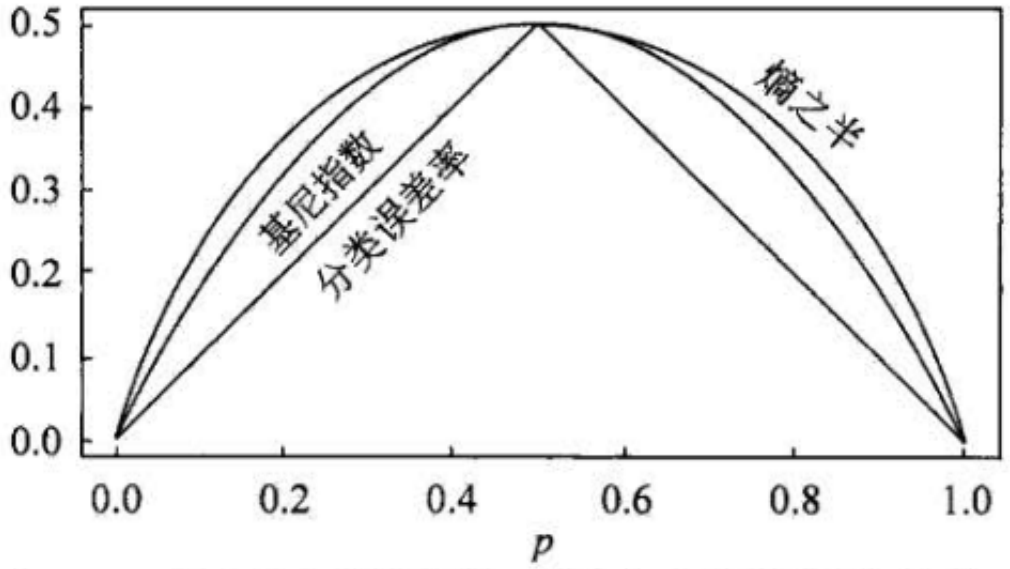


图 5.7　二类分类中基尼指数、熵之半和分类误差率的关系

三、决策树的生成

3.1 ID3算法

ID3算法的核心是在决策树各个结点上应用信息增益准则选择特征，递归地建构决策树。

其具体方法为：从根结点开始，对结点计算所有可能的特征的信息增益，选择信息增益最大的特征作为结点的特征，由该特征的不同取值建立子结点；再对子结点递归地调用以上方法，构建决策树；直到所有特征的信息增益均很小或没有特征可以选择为止。最后得到一个决策树。ID3相当于用极大似然法进行概率模型的选择。但是ID3算法只有树的生成，所以该算法生成的树容易产生过拟合。

其算法步骤如下：

- 1） 若D中所有实例属于同一类C_k，则T为单结点树，并将类C_k作为该结点的类标记，返回T;
- 2）若$A = \varnothing$,则T为单结点树，并将D中实例数最大的类C_k作为该结点的类标记，返回T;
- 3） 否则，按算法5.1计算A中各特征对D的信息增益，选择信息增益最大的特征A_g;
- 4） 如果A_g的信息增益小于阈值ε，则置T为单结点树，并将D中实例数最大的类C_k作为该结点的类标记，返回T
- 5） 否则，对A_g的每一个可能值a_i，依$A_g = a_i$将D分割为若干非空子集D_i,将D_i中实例数最大的类作为标记，构建子结点，由结点及其子节点构成树T,返回T

- 6） 对第i个子结点，以D_i为训练集,以$A-A_g$为特征集，递归地调用（1）~（5），得到子树T_i，返回T。

3.2 C4.5

与ID3算法相似，C4.5算法对ID3算法进行了改进，C4.5在生成的过程中，用信息增益比来选择特征

3.3 CART

分类树与回归树（classification and regression tree，CART）模型（Breiman）由特征选择、树生成及剪枝组成，既可用于分类也可用于回归。CART是在给定输入随机变量X条件下输出变量Y的条件概率分布的学习方法。它假定决策树是二叉树，内部取值为“是”（左分支）和“否”（右分支）。
它的基本步骤为

- 1）决策树生成：基于训练数据集生成决策树，生成的决策树要尽量大。
- 2）决策树剪枝：用验证数据集对已生成的树进行剪枝并选择最优子树，这是用损失函数最小作为剪枝的标准。

3.3.1 分类树

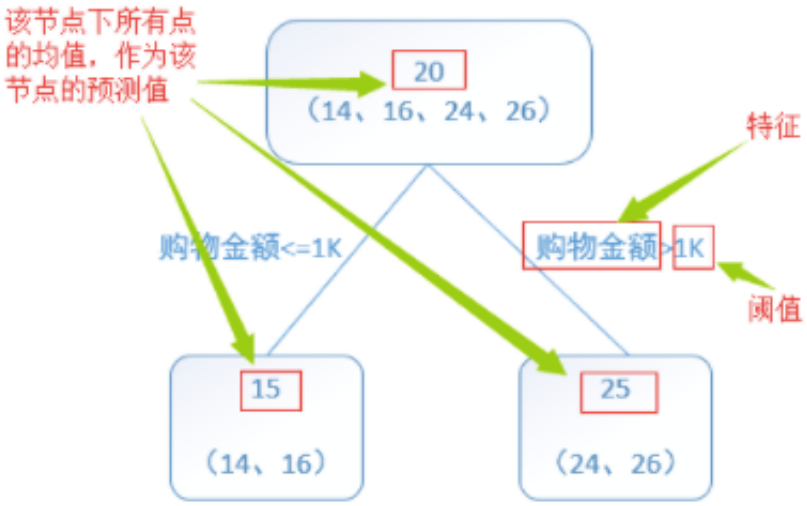
对分类树用基尼系数（Gini index）最小化准则，进行特征选择，生成二叉树。

具体算法步骤如下：

- 1）设结点的训练数据集为D，计算现有特征对该数据集的基尼指数。此时，对每一个特征A，对其可能取的每个值a，根据样本点对$A=a$的测试为"是"或者“否”将D分割为D_1和D_2两部分，计算其基尼系数。
- 2）在所有可能的特征A以及他们所有可能的切分点a中，选择基尼系数最小的特征及其对应的切分点作为最优特征与最优切分点。依最优特征与最优切分点，从现结点生成两个子结点，将训练数据集依特征分配到两个子结点中去。
- 3）对两个子结点递归地调用上述两个步骤，直至满足停止条件。
- 4）生成CART决策树

3.3.2 回归树

首先看一个简单的回归树生成实例：



接下来具体说说回归树是如何进行特征选择生成二叉回归树的。
假设X与Y分别为输入和输出变量，并且Y是连续变量，给定训练数据集

$$D = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

我们利用最小二乘回归树生成算法来生成回归树$f(x)$，即在训练数据集所在的输入空间中，递归地将每个区域分为两个子区域并决定每个子区域上的输出值，构建二叉决策树，步骤如下：

- 1）选择最优切分变量j与切分点s，求解

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}\left(y_i-c_1\right)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}\left(y_i-c_2\right)^2\right]$$

  遍历变量j，对固定的切分变量j扫描切分点s，选择使上式达到最小值得对j, s
- 2）用选定的对(j, s)划分区域并决定相应的输出值：

$$R_1(j,s)=\left\{x\,|\,x^{(j)}\le s\right\},\ R_2(j,s)=\left\{x\,|\,x^{(j)}>s\right\}$$

$$\hat{c}_m=\frac{1}{N_m}\sum_{x_i\in R_2(j,s)}y_i,\ x\in R_m,\ m=1,2$$

- 3）继续对两个子区域调用步骤（1），（2），直至满足停止条件。
- 4）将输入空间划分为M个区域$R_1, R_2, \cdots, R_M$，在每个单元R_m上有一个固定的输出值c_m，生成决策树：

$$f(x)=\sum_{m=1}^{M}\hat{c}_m\mathrm{I}\left(\mathrm{x}\in\mathrm{R_m}\right)$$

四、决策树的剪枝

4.1 剪枝

决策树的过拟合指的是学习时过多地考虑如何提高对训练数据的正确分类，从而构建出过于复杂的决策树。解决过拟合的办法是考虑决策树的复杂度，对已生成的决策树进行简化，即剪枝（从已生成的树上裁剪调一些子树或叶结点，并将其根结点或父结点作为新的叶结点，从而简化分类树模型）。

设树T的叶结点个数为$|T|$,t是树T的叶结点有N_t个样本点，其中k类的样本点有N_{tk}个，$k=1,2,\cdots,K$，$H_t(T)$为叶结点t上的经验熵，$a \geq 0$为参数，则决策树学习的损失函数可以定义为

$$C_a(T)=\sum_{t=1}^{|T|}N_tH_t(T)+a|T|$$

其中经验熵为

$$H_t(T)=-\sum_k\frac{N_{tk}}{N_t}log\frac{N_{tk}}{N_t}$$

在损失函数中，将右端第一项记作

$$C(T)=\sum_{t=1}^{|T|}N_tH_t(T)=-\sum_{t=1}^{|T|}\sum_{k=1}^{K}N_{tk}\log\frac{N_{tk}}{N_t}$$

这时有

$$C_a(T)=C(T)+a|T|$$

其中，$C(T)$表示模型对训练数据的预测误差,即模型与训练数据的拟合程度，$|T|$表示模型复杂度，参数$a \geq 0$控制两者之间的影响。较大的a促使选择较简单的模型，较小的a促使选择较复杂的模型。$a=0$意味着只考虑模型与训练数据的拟合程度，不考虑模型的复杂度。

决策树生成只考虑了通过信息增益（或信息增益比）对训练数据进行更好的拟合。而决策树剪枝通过优化损失函数还考虑了减小模型复杂度。决策树生成学习局部的模型，而决策树剪枝学习整体的模型。此损失函数的极小化等价于正则化的极大似然估计，即利用损失函数最小原则进行剪枝就是用正则化的极大似然估计进行模型选择。

4.2 CART剪枝

CART剪枝算法从“完全生长”的决策树的底端减去一些子树，使决策树变小（模型变简单），从而能够对未知数据有更准确的预测
其具体步骤如下：

- 1）首先从生成算法产生的决策树T_0底端开始不断剪枝，直到T_0的根节点，形成一个字数序列$\left\{T_0, T_1, T_{2,.....,} T_n\right\}$;

在剪枝过程中，计算子树的损失函数：

$$C_a(T) = C(T) + a|T|$$

其中，T为任意子树，$C(T)$为对训练数据的预测误差（如基尼系数），$|T|$为子树的叶结点个数，$a \geq 0$为参数，$C_a(T)$为参数是a时的子树T的整体损失。参数a权衡训练数据的拟合程度与模型的复杂度。

对固定的a，一定存在使损失函数$C_a(T)$最小的子树，将其表示为T_a。T_a在损失函数$C_a(T)$最小的意义下是最优的，且是唯一的。a大的时候，最优子树T_a偏小；当a小的时候，最优子树T_a偏大。极端情况，$a = 0$时，整体树是最优的。当$a \rightarrow \infty$，根结点组成的单结点树是最优的。

Breiman等人证明：可以用递归地方法对树进行剪枝。将a从小增大，
$0 = a_0 < a_1 < .....a_n < +\infty$产生一系列的区间$[a_i, a_{i+1}), i = 0, 1, ..., n$;剪枝得到的子树序列对应着区间$a \in \left[a_i, a_{i+1}\right)$，$i = 0, 1, 2, ..., n$的最优子树序列为$\left\{T_0, T_1, T_2, ..., T_n\right\}$，序列的子树是嵌套的。

具体地，从整体树T_0开始剪枝，对T_0的人以内部结点t，以t为单结点树的损失函数是

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

以t为根结点的子树T_t的损失函数是

$$C_a(T_t) = C(T_t) + a|T_t|$$

当$a = 0$及a充分小时，有不等式

$$C_a\left(T_t\right) 小 \ 于 \ C_a(T)$$

当a增大时，在某一a有

$$C_a(T_t) \text{ 等 于 } C_a(t)$$

当a再增大时，有不等式

$$C_a(T_t) \text{ 大 于 } C_a(T)$$

只要$\alpha = \frac{C(t) - C\left(T_t\right)}{|T_t| - 1}$，$T_t$与$t$有相同的损失函数值，而$t$的结点少，因此$t$比$T_t$更可取，对$T_t$进行剪枝。

为此，对T_0中的每一个内部结点t，计算

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

它表示剪枝后整体损失函数减少的程度。在T_0中剪去$g(t)$最小的T_t，将得到的子树作为T_1，同时将最小的$g(t)$设为a_1，T_1为区间$[a_1, a_2)$的最优子树。

如此剪枝下去，直至得到根结点。在这一过程中，不断得增加a的值，产生新的区间。

- 2）在剪枝得到的子树序列$T_0, T_1, \ldots, T_n$中通过交叉验证选取最优子树T_a

具体地，利用独立的验证数据集，测试子树序列$T_0, T_1, \ldots, T_n$中各棵子树的平方误差或基尼指数。平方误差或基尼指数最小的决策树被认为是最优的决策树。在子树序列中，每棵子树$T_0, T_1, \ldots, T_n$都对应一个参数$a_1, a_2, \ldots, a_n$。所以当最优子树T_k确定时，对应的a_k也就确定了，即得到最由决策树T_a.

五、参考资料

李航《统计学习方法》

周志华《机器学习》