统计学习方法

第五章　决策树

颜子彦

2018.11.02

第五章 决策树

3.1 基本概念

- 决策树是一种基本的分类和回归方法
- 简单描述：
  - 问题：对一个类别未知的样本点A进行分类
  - 方法：
    - 从一个已知分类的数据集中，找出距离A最近的k个点
    - 依据这k个点的大多数类别，作为A的类别

3.1 基本概念

- 决策树模型
  - 决策树由结点和有向边组成。
  - 结点分两种类型：内部结点和叶结点。内部结点表示一个特征或属性，叶结点表示一个类
  - 根据测试结果将实例分配到其子结点。如此递归的对实例测试并分配，直至达到叶结点。最后将实例分到叶结点的类中。

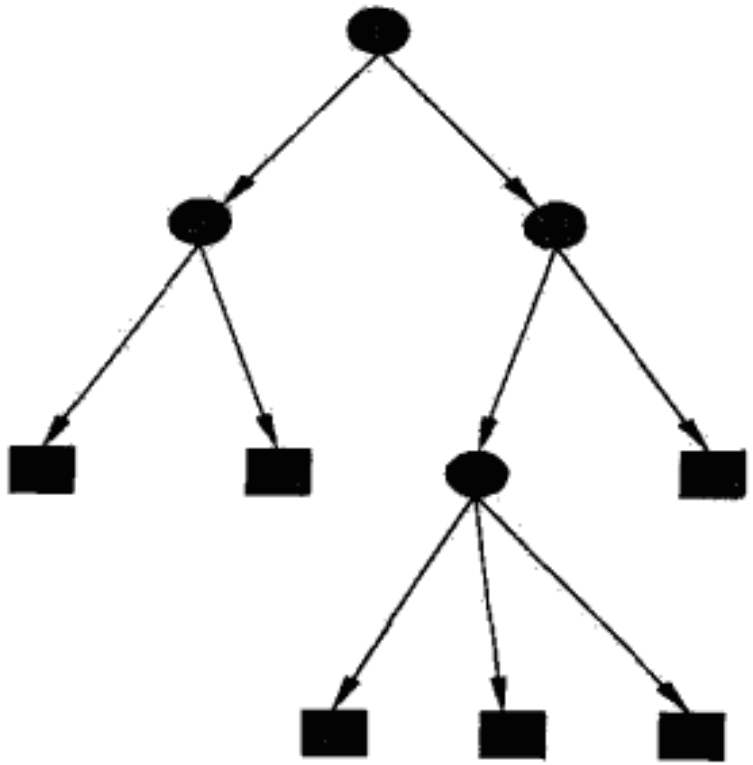
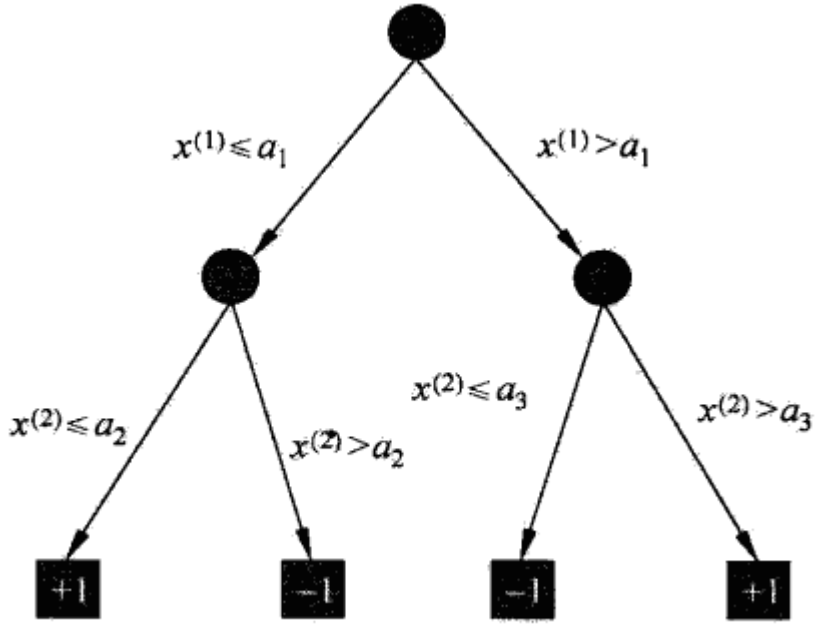


图 5.1　决策树模型

3.1 基本概念

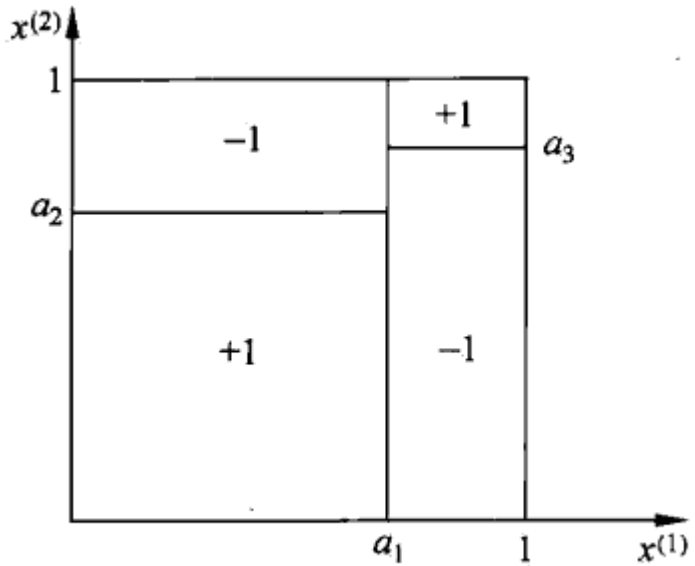
- 决策树与if-then规则
  - 可将决策树看成一个if-then规则的集合。
  - 每一个实例都被一条路径或一条规则所覆盖，而且只被一条路径或一条规则所覆盖。



(c) 决策树

3.1 基本概念

- 决策树与条件分布
  - 决策树还表示给定特征条件下类的条件概率分布。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，那么条件概率分布可以表示为P（Y|X）。



(a) 特征空间划分

3.1 基本概念

- 决策树学习
  - 决策树学习的损失函数通常是正则化的极大似然函数。
  - 决策树学习的策略是以损失函数为目标函数的最小化。
  - 当损失函数确定以后，学习问题就变为在损失函数意义下选择最优决策树的问题。
  - 决策树学习的算法通常是递归地选择最优特征，并根据特征对训练数据进行分割。

3.1 基本概念

- 决策树学习
  - 生成的决策树可能对训练数据有很好的分类能力，但对未知的测试数据未必有好的分类能力，即可能发生过拟合现象。
  - 决策树的生成只考虑局部最优，决策树的剪枝则考虑全局最优。
  - 决策树学习常用的算法有ID3、C4.5、CART。

第五章 决策树

3.2 特征的选择

表 5.1 贷款申请样本数据表

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

3.2 特征的选择

- 特征选择是决定用哪个特征来划分特征空间。
- 直观上，如果按照一个特征将训练数据集分割成子集，使得各个子集在当前条件下有最好的分类，那么久应该选择这个特征。
- 信息增益就能够很好的表示这一直观的准则。

3.2 特征的选择

- 熵
  - 熵表示随机变量不确定性的度量。
  - 设X是一个取有限个值的离散随机变量，其概率分布为$P(X = x_i) = p_i,\ i = 1,2,\dots,n$
  - 则随机变量X的熵定义为$H(X) = -\sum_{i=1}^{n} p_i \log p_i$
  - 定义0log0=0

3.2 特征的选择

- 条件熵
  - 熵越大，随机变量的不确定性就越大。
  - 条件熵表示在已知X的条件下Y的不确定性
  - 设随机变量（X，Y），其联合概率分布为$P(X = x_i, Y = y_i) = p_{ij},\ i = 1,2,\dots,n; j = 1,2,\dots,m$
  - 则随机变量X给定的条件下随机变量Y的条件熵定义为$H(Y|X) = \sum_{i=1}^{n} p_i H(Y|X = x_i)$
  - 这里，$p_i = P(X = x_i),\ i = 1,2,\dots,n$

3.2 特征的选择

- 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。
  - $g(D,A)=H(D)-H(D\mid A)$
  - 显然，对于数据集而言，信息增益依赖于特征，不同的特征具有不同的信息增益。
  - 对训练数据集D，计算其每个特征的信息增益，选择信息增益最大的特征。

3.2 特征的选择

- 信息增益比
  - 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的值的熵之$H_A(D)$比，即$g_R(D,A)=\frac{g(D,A)}{H_A(D)}$。
  - $g(D,A)=H(D)-H(D\mid A)$
  - 其中，$H_A(D)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\log_2\frac{|D_i|}{|D|}$，n是特征A取值的个数。
  - 对训练数据集D，计算其每个特征的信息增益，选择信息增益最大的特征。

第五章 决策树

3.3 决策树的生成

- ID3算法
  - 输入：训练数据集D，特征集A，阈值 ε 。
  - 输出：决策树T

（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=∅，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；

3.3 决策树的生成

- ID3算法

（3）否则，计算5.1计算A中个特征对D的信息增益，选择信息增益最大的特征A_g

（4）若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，由结点及其子结点构成树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_i。

3.3 决策树的生成

- C4.5算法
  - 输入：训练数据集D，特征集A，阈值 ε 。
  - 输出：决策树T

（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=∅，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；

3.3 决策树的生成

- C4.5算法

（3）否则，计算5.1计算A中个特征对D的信息增益比，选择信息增益最大的特征A_g

（4）若A_g的信息增益比小于阈值 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，由结点及其子结点构成树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_i。

3.3 决策树的生成

Day	Temperatrue	Outlook	Humidity	Windy	PlayGolf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	sunny	high	true	no
07-26	cool	sunny	normal	true	no
07-30	mild	sunny	high	false	yes

第五章 决策树

3.4 决策树的剪枝

- 过拟合的原因：构建出过于复杂的决策树。

- 决策树的剪枝往往通过极小化决策树整体的损失函数或代价函数来实现

3.4 决策树的剪枝

- 决策树的损失函数
  - 决策树学习的损失函数可以定义为$C_a(T)=\sum_{t=1}^{|T|}N_tH_t(T)+\alpha|T|$
  - 其中经验熵为$H_t(T)=-\sum_k\frac{N_{tk}}{N_t}\log\frac{N_{tk}}{N_t}$
  - 记$C(T)=\sum_{t=1}^{|T|}N_tH_t(T)=-\sum_{t=1}^{|T|}\sum_{k=1}^{K}N_{tk}\log\frac{N_{tk}}{N_t}$
  - 这时有$C_a(T)=C(T)+\alpha|T|$

3.4 决策树的剪枝

- 树的剪枝算法
  - 输入：生成算法产生的整个树T，参数 α 。
  - 输出：修建后的子树T_α

  （1）计算每个结点的经验熵；

  （2）递归地从树的叶结点向上回缩；

  （3）返回（2），直至不能继续位置，得到损失函数最小的子树T_α ；

第五章 决策树

3.5 CART算法

- CART算法有以下两步组成：
  - 决策树生成：生成的决策树尽量大
  - 决策树剪枝：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。

3.5 CART算法

- CART生成：
  - 对回归树用平方误差最小化准则
  - 当输入空间的划分确定时，可以用平方误差$\sum_{x_i \in R_m}(y_i - f(x_i))^2$来表示回归树对于训练数据的预测误差。

3.5 CART算法

- CART回归树的生成：
  - 选择第j个变量$x^{(j)}$和它取的值s，作为切分变量和切分点并定义两个区域
  - $R_1(j,s)=\{x|x^j \le s\}$和$R_2(j,s)=\{x|x^j \ge s\}$
  - 然后求解$\min_{j,s}[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2]$
  - 遍历所有输入变量，找到最优的切分变量j，构成一个对（j，s）依此划分为两个区域。
  - 接着重复上述过程直到满足停止条件为止。这样就生成一棵回归树，这样的回归树通常称为最小二乘回归树。

3.5 CART算法

- CART回归树的生成：

算法 5.5（最小二乘回归树生成算法）

输入：训练数据集 D；

输出：回归树 $f(x)$.

在训练数据集所在的输入空间中，递归地将每个区域划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：

（1）选择最优切分变量 j 与切分点 s，求解

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right] \tag{5.21}$$

遍历变量 j，对固定的切分变量 j 扫描切分点 s，选择使式 (5.21) 达到最小值的对 (j,s).

（2）用选定的对 (j,s) 划分区域并决定相应的输出值：

$$R_1(j,s)=\{x\mid x^{(j)}\leqslant s\},\quad R_2(j,s)=\{x\mid x^{(j)}>s\}$$

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

（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 $R_1,R_2,\cdots,R_M$，生成决策树：

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

■

3.5 CART算法

- CART生成：
  - 对分类树用基尼指数最小化准则
  - 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，则概率分布的基尼指数定义为$Gini(p) = \sum_{k=1}^{K} p_k(1-p_k) = 1 - \sum_{k=1}^{K} {p_k}^2$

3.5 CART算法

- 基尼指数：
  - 对于二分类问题，若样本点属于第一个类的概率为p，则概率分布的基尼指数为$Gini(p) = 2p(1-p)$
  - 对于给定的样本集合D，其基尼指数为$Gini(D) = 1 - \sum_{k=1}^{K}(\frac{|C_k|}{|D|})^2$
  - 这里C_k是D中属于第k类的样本子集，K是类的个数

3.5 CART算法

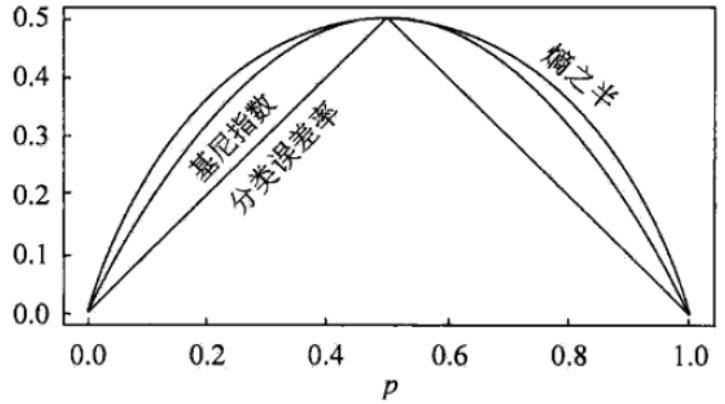


图 5.7　二类分类中基尼指数、熵之半和分类误差率的关系

- 基尼指数：
  - 如果样本集合D根据特征A是否取某一可能值a被分割成D_1和D_1两部分，即 $D_1 = \{(x, y) \in D | A(x) = a\}, D_2 = D - D_1$
  - 则在特征A的条件下，集合D的基尼指数定义为

    $$Gini(D, A) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2)$$
  - 基尼指数表示集合D的不确定性。基尼指数值越大，样本集合的不确定性也就越大，这一点与熵类似

3.5 CART算法

算法 5.6（CART 生成算法）

输入：训练数据集 D，停止计算的条件；

输出：CART 决策树.

根据训练数据集，从根结点开始，递归地对每个结点进行以下操作，构建二叉决策树：

（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“否”将 D 分割成 D_1 和 D_2 两部分，利用式（5.25）计算 $A=a$ 时的基尼指数.

（2）在所有可能的特征 A 以及它们所有可能的切分点 a 中，选择基尼指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（1），（2），直至满足停止条件.

（4）生成 CART 决策树. ■

算法停止计算的条件是结点中的样本个数小于预定阈值，或样本集的基尼指数小于预定阈值（样本基本属于同一类），或者没有更多特征.

3.5 CART算法

- CART剪枝：
  - 子树的损失函数$C_\alpha(T) = C(T) + \alpha|T|$，其中T为任意子树，$C(T)$为对训练数据的预测误差，参数 α 权衡训练数据的拟合程度与模型的复杂度
  - 当 α 大时，最优子树偏小
  - 当 α 小时，最优子树偏大
  - 可以用递归地方法对树进行剪枝，产生一系列的区间，剪枝得到的子树序列与之一一对应
  - 序列中的子树是嵌套的

3.5 CART算法

- CART剪枝：
  - 对T_0的任意内部结点t，以t为单节点树的损失函数为$C_a(t) = C(t) + \alpha$
  - 以t为根结点的子树T_t的损失函数是$C_a(T_t) = C(T_t) + \alpha|T_t|$
  - α充分小时，有$C_a(T_t) < C_a(t)$
  - 随着α增大时，在某一α有$C_a(T_t) = C_a(t)$
  - 只要$\alpha = \frac{C(t)-C(T_t)}{|T_t|-1}$，$T_t$与t有相同的损失函数值，而t的节点少，因此t更可取，进行剪枝。
  - 为此，对T_0中每一内部结点t，计算$g(t) = \frac{C(t)-C(T_t)}{|T_t|-1}$表示剪枝后整体损失函数减少的程度。

3.5 CART算法

算法 5.7（CART 剪枝算法）

输入：CART 算法生成的决策树 T_0；

输出：最优决策树 T_α.

（1）设 $k=0$，$T=T_0$.

（2）设 $\alpha=+\infty$.

（3）自下而上地对各内部结点 t 计算 $C(T_t)$，$|T_t|$ 以及

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

$$\alpha=\min(\alpha,g(t))$$

这里，T_t 表示以 t 为根结点的子树，$C(T_t)$ 是对训练数据的预测误差，$|T_t|$ 是 T_t 的叶结点个数.

（4）自上而下地访问内部结点 t，如果有 $g(t)=\alpha$，进行剪枝，并对叶结点 t 以多数表决法决定其类，得到树 T.

（5）设 $k=k+1$，$\alpha_k=\alpha$，$T_k=T$.

（6）如果 T 不是由根结点单独构成的树，则回到步骤 (4).

（7）采用交叉验证法在子树序列 $T_0,T_1,\cdots,T_n$ 中选取最优子树 T_α. ■

3.5 CART算法

算法	支持模型	树结构	特征选择	连续值处理	缺失值处理	剪枝
ID3	分类	多叉树	信息增益	不支持	不支持	不支持
C4.5	分类	多叉树	信息增益比	支持	支持	支持
CART	分类，回归	二叉树	基尼系数，均方差	支持	支持	支持

决策树的优点

- 1）简单直观，生成的决策树很直观。
- 2）基本不需要预处理，不需要提前归一化，处理缺失值。
- 3）使用决策树预测的代价是O(log2m)O(log2m)。 m为样本数。
- 4）既可以处理离散值也可以处理连续值。很多算法只是专注于离散值或者连续值。
- 5）可以处理多维度输出的分类问题。
- 6）相比于神经网络之类的黑盒分类模型，决策树在逻辑上可以得到很好的解释
- 7）可以交叉验证的剪枝来选择模型，从而提高泛化能力。
- 8） 对于异常点的容错能力好，健壮性高。

决策树的缺点

- 1）决策树算法非常容易过拟合，导致泛化能力不强。可以通过设置节点最少样本数量和限制决策树深度来改进。
- 2）决策树会因为样本发生一点点的改动，就会导致树结构的剧烈改变。这个可以通过集成学习之类的方法解决。
- 3）寻找最优的决策树是一个NP难的问题，我们一般是通过启发式方法，容易陷入局部最优。可以通过集成学习之类的方法来改善。
- 4）有些比较复杂的关系，决策树很难学习，比如异或。这个就没有办法了，一般这种关系可以换神经网络分类方法来解决。
- 5）如果某些特征的样本比例过大，生成决策树容易偏向于这些特征。这个可以通过调节样本权重来改善。