# 第 5 章　决策树

决策树是一种基于规则的方法，它用一组嵌套的规则进行预测。在树的每个决策节点处，根据判断结果进入一个分支，反复执行这种操作直到到达叶子节点，得到预测结果。这些规则通过训练得到，而不是人工制定的。

## 5.1　树形决策过程

首先看一个简单的例子。银行要确定是否给客户发放贷款，为此需要考察客户的收入与房产情况。在做决策之前，会先获取客户的这两个数据。如果把这个决策看作分类问题，两个指标就是特征向量的两个分量，分类的类别标签是可以贷款和不能贷款。银行按照下面的过程进行决策：

1.首先判断客户的年收入指标。如果大于 20 万，可以贷款；否则继续判断。

2.然后判断客户是否有房产。如果有房产，可以贷款；否则不能贷款。

用图形表示这个过程就是一棵决策树。决策过程从树的根节点开始，在内部节点处需要做判断，直到到达一个叶子节点处，得到决策结果。决策树由一系列分层嵌套的判定规则组成，是一个递归的结构。这个例子的决策树如下图 5.1 所示：

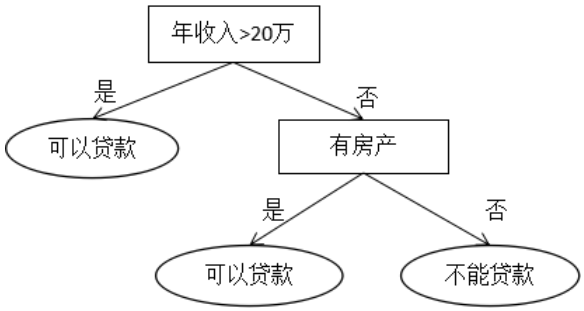


图 5.1　决策树的一个例子

收入为数值型特征，可以比较大小，这种特征为整数或小数。房产情况为类别型特征，取值为有房产和没有房产两种情况，这种特征不能比较大小。上图中决策树所有的内部节点为矩形，叶子节点即决策结果为椭圆形。

为便于用程序实现，一般将决策树设计成二叉树。与树的叶子节点、非叶子节点相对应，决策树的节点分为两种类型：

决策节点。在这些节点处需要进行判断以决定进入哪个分支，如用一个特征和设定的阈值进行比较。决策节点一定有两个子节点，它是非叶子节点。

叶子节点。表示最终的决策结果，它们没有子节点。在上面的例子中，叶子节点的值有两种，即能贷款和不能贷款。对于分类问题，叶子节点中存储的是类别标签。

决策树是一个分层结构，可以为每个节点赋予一个层次数。根节点的层次数为 0，子节点的层次数为父节点层次数加 1。树的深度定义为所有节点的最大层次数。上图决策树的深度为 2，要得到一个决策结果最多经过 2 次判定。

典型的决策树有 ID3[1]，C4.5[3]，CART（Classification and Regression Tree，分类与回

归树）[4]等，它们的区别在于树的结构与构造算法。分类与回归树既支持分类问题，也可用于回归问题。决策树是一种判别模型，天然支持多类分类问题。限于篇幅，本章只介绍分类与回归树。

分类树的映射函数是多维空间的分段线性划分，即用平行于各坐标轴的超平面对空间进行切分；回归树的映射函数是分段常数函数。决策树是分段线性函数而不是线性函数，它具有非线性建模的能力。只要划分的足够细，分段常数函数可以逼近闭区间上任意函数到任意指定精度，因此决策树在理论上可以对任意复杂度的数据进行拟合。对于分类问题，如果决策树深度够大，它可以将训练样本集的所有样本正确分类。但如果特征向量维数过高，可能会面临维数灾难导致准确率下降，维数灾难的概念将在第 14 章介绍。

下图 5.2 是决策树对空间划分的一个例子。这里有红色和蓝色两类训练样本，用下面两条平行于坐标轴的直线可以将这两类样本分开：

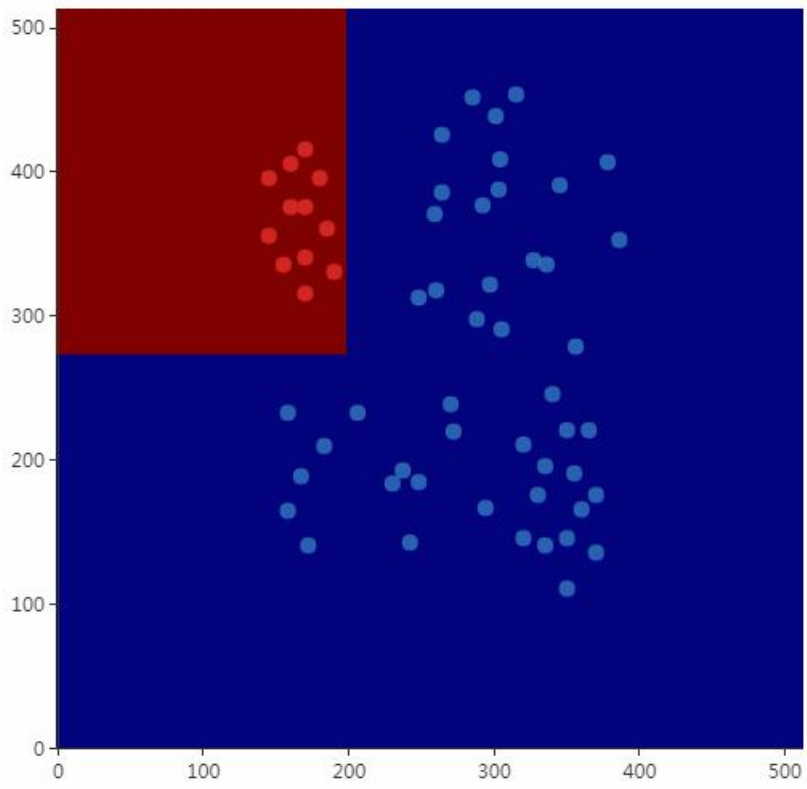


图 5.2　决策树对空间的划分

这个划分方案对应的决策树如下图 5.3 所示：

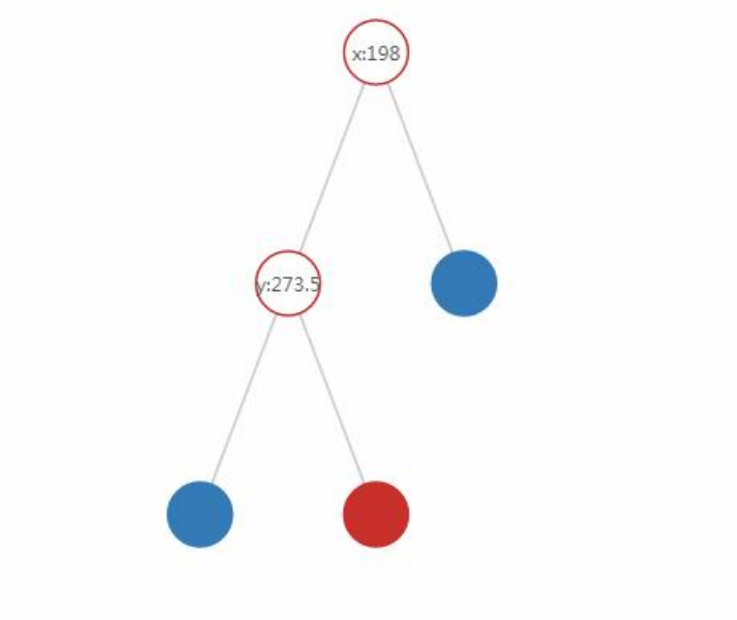


图 5.3　分类树

## 5.2　分类与回归树

下面介绍分类与回归树的原理，它是二叉决策树。预测时从根节点开始，每次只对一个特征进行判定然后进入左子节点或者右子节点，直至到达一个叶子节点处，得到类别值或回归函数值。预测算法的时间复杂度与树的深度有关，判定的执行次数不超过决策树的深度。

## 5.3　训练算法

现在要解决的关键问题是如何用训练样本建立决策树。无论是分类问题还是回归问题，决策树都要尽可能的对训练样本进行正确的预测。直观的想法是从根节点开始构造，递归的用训练样本集建立起决策树，这棵树能够将训练集正确的分类，或者对训练集的回归误差最小化。为此要解决以下问题：

特征向量有多个分量，每个决策节点上应该选择哪个分量做判定？这个判定会将训练样本集一分为二，然后用这两个子集构造左右子树。

在选定一个特征后，判定的规则是什么？即满足什么条件时进入左子树分支。对数值型变量要寻找一个分裂阈值进行判断，小于该阈值进入左子树，否则进入右子树。对于类别型变量则需要为它确定一个子集划分，将特征的取值集合划分成两个不相交的子集，如果特征的值属于第一个子集则进入左子树，否则进入右子树。

何时停止分裂，把节点设置为叶子节点？对于分类问题，当节点的样本都属于同一类型时停止，但这样可能会导致树的节点过多、深度过大，产生过拟合问题。另一种方法是当节点中的样本数小于一个阀值时停止分裂。

如何为每个叶节点赋予类别标签或者回归值？即到达叶子节点时样本被分为哪一类或者赋予一个什么实数值。

下面给出这几个问题的答案。特征有数值型变量和类别型变量两种情况，决策树有分类树和回归树两种类型，组合起来一共有 4 种情况。数值型变量是指能够比较大小的变量，如年龄，速度；类别型变量是整数编码，不能比较大小，如各种类型的动物。限于篇幅，我们

只对数值型变量进行介绍。

### 5.3.1 递归分裂过程

训练算法是一个递归的过程。首先创建根节点，然后递归的建立左子树和右子树。如果练样本集为 $D$，训练算法的整体流程为：

1.用样本集 $D$ 建立根节点，找到一个判定规则，将样本集分裂成 $D_1$ 和 $D_2$ 两部分，同时为根节点设置判定规则。

2.用样本集 $D_1$ 递归建立左子树。

3.用样本集 $D_2$ 递归建立右子树。

4.如果不能再进行分裂，则把节点标记为叶子节点，同时为它赋值。

在确定这个递归流程之后，接下来要解决的核心问题是怎样对训练样本集进行分裂。

### 5.3.2 寻找最佳分裂

训练时需要找到一个分裂规则把训练样本集分裂成两个子集，因此要确定分裂的评价标准，根据它寻找最佳分裂。对于分类问题，要保证分裂之后左右子树的样本尽可能的纯，即它们的样本尽可能属于不相交的某一类或者几类。为此需要定义不纯度的指标：当样本都属于某一类时不纯度为 0；当样本均匀的属于所有类时不纯度最大。满足这个条件的有熵不纯度，Gini 不纯度，以及误分类不纯度等，下面分别进行介绍。

不纯度指标用样本集中每类样本出现的概率值构造。因此首先要计算每个类出现的概率，这通过训练样本集中每类样本数除以样本总数得到

$$p_i = \frac{N_i}{N}$$

其中 $N_i$ 是第 $i$ 类样本数， $N$ 为总样本数。根据这个概率值可以定义各种不纯度指标，下面分别介绍。

样本集 $D$ 的熵不纯度定义为

$$E(D) = -\sum_i p_i \log_2 p_i$$

熵是信息论中的一个重要概念，用来度量一组数据包含的信息量大小。当样本只属于某一类时熵最小，当样本均匀的分布于所有类中时熵最大。因此，如果能找到一个分裂让熵最小，这就是我们想要的最佳分裂。

样本集的 Gini 不纯度定义为

$$G(D) = 1 - \sum_i p_i^2$$

当样本属于某一类时 Gini 不纯度的值最小，此时最小值为 0；当样本均匀的分布与每一类时 Gini 不纯度的值最大。这源自于如下的数学结论，在下面的约束条件下：

$$\sum_i p_i = 1$$

$$p_i \geq 0$$

对于如下目标函数：

$$\sum_i p_i^2$$

所有变量相等时它有极小值，只有一个变量为 1 其他变量为 0 时该函数有极大值，这对应于 Gini 不纯度的极小值。这一结论可以通过拉格朗日乘数法证明。即所有样本都来自同一类时 Gini 不纯度的值最小，样本均匀的属于每一类时 Gini 不纯度的值最大。将类概率的计算公式代入 Gini 不纯度的定义，可以得到简化的计算公式

$$G(D)=1-\sum_i p_i^2=1-\sum_i (N_i/N)^2=1-\left(\sum_i N_i^2\right)/N^2$$

样本集的误分类不纯度定义为

$$E(D)=1-\max(p_i)$$

之所以这样定义是因为我们会把样本判定为频率最大的那一类，因此其他样本都会被错分，故错误分类率为上面的值。和上面的两个指标一样，当样本只属于某一类时误分类不纯度有最小值 0，样本均匀的属于每一类时该值最大。

上面定义的是样本集的不纯度，我们需要评价的是分裂的好坏，因此需要根据样本集的不纯度构造出分裂的不纯度。分裂规则将节点的训练样本集分裂成左右两个子集，分裂的目标是把数据分成两部分之后这两个子集都尽可能的纯，因此我们计算左右子集的不纯度之和作为分裂的不纯度，显然求和需要加上权重，以反映左右两边的训练样本数。由此得到分裂的不纯度计算公式为

$$G=\frac{N_L}{N}G(D_L)+\frac{N_R}{N}G(D_R)$$

其中 $G(D_L)$ 是左子集的不纯度，$G(D_R)$ 是右子集的不纯度，$N$ 是总样本数，$N_L$ 是左子集的样本数，$N_R$ 是右子集的样本数。

如果采用 Gini 不纯度指标，将 Gini 不纯度的计算公式代入上式可以得到

$$\begin{aligned}G&=\frac{N_L}{N}\left(1-\frac{\sum_i N_{L,i}^2}{N_L^2}\right)+\frac{N_R}{N}\left(1-\frac{\sum_i N_{R,i}^2}{N_R^2}\right)\\&=\frac{1}{N}\left(N_L-\frac{\sum_i N_{L,i}^2}{N_L}+N_R-\frac{\sum_i N_{R,i}^2}{N_R}\right)\\&=1-\frac{1}{N}\left(\frac{\sum_i N_{L,i}^2}{N_L}+\frac{\sum_i N_{R,i}^2}{N_R}\right)\end{aligned}$$

其中 $N_{L,i}$ 是左子节点中第 $i$ 类样本数，$N_{R,i}$ 是右子节点中第 $i$ 类样本数。由于 $N$ 是常数，要让 Gini 不纯度最小化等价于让下面的值最大化

$$G=\frac{\sum_i N_{L,i}^2}{N_L}+\frac{\sum_i N_{R,i}^2}{N_R}$$

这个值可以看做是 Gini 纯度，它的值越大，样本越纯。寻找最佳分裂时需要计算用每

个阈值对样本集进行分裂后的这个值，寻找该值最大时对应的分裂，它就是最佳分裂。这比直接按照 Gini 系数的定义计算的计算量更小，避免了大量的浮点数运算。如果是数值型特征，对于每个特征将 $l$ 个训练样本按照该特征的值从小到大排序，假设排序后的值为

$$x_1,...,x_l$$

接下来从 $x_1$ 开始，依次用每个 $x_i$ 作为阈值，将样本分成左右两部分，计算上面的纯度值，该值最大的那个分裂阈值就是此特征的最佳分裂阈值。在计算出每个特征的最佳分裂阈值和上面的纯度值后，比较所有这些分裂的纯度值大小，该值最大的分裂为所有特征的最佳分裂。这里采用了贪心法的策略，每次都是选择当前条件下最好的分裂作为当前节点的分裂。对单个变量寻找最佳分裂阈值的过程如下图 5.4 所示：

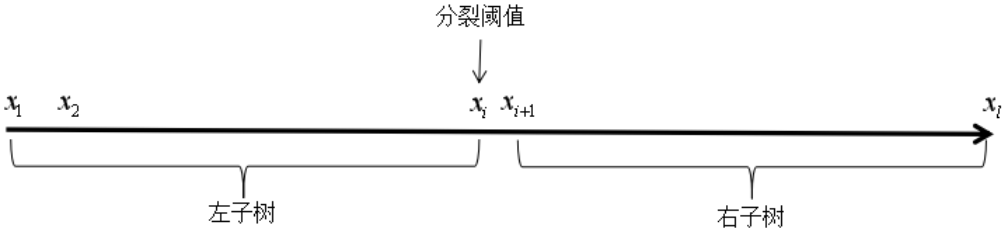


图 5.4　为数值型变量寻找最佳分裂阈值

对于回归树，衡量分裂的标准是回归误差即样本方差，每次分裂时选用使得方差最小化的那个分裂。假设节点的训练样本集有 $l$ 个样本 $\left(\mathbf{x}_i, y_i\right)$，其中 $\mathbf{x}_i$ 为特征向量，$y_i$ 为实数的标签值。节点的回归值为所有样本的均值，回归误差为所有样本的标签值与回归值的均方和误差，定义为

$$E(D)=\frac{1}{l}\sum_{i=1}^{l}\left(y_i-\bar{y}\right)^2$$

可以证明，回归值为均值的时候，上面的均方误差最小。把均值的定义带入上式，得到

$$\begin{aligned}E(D)&=\frac{1}{l}\sum_{i=1}^{l}\left(y_i-\frac{1}{l}\sum_{j=1}^{l}y_j\right)^2\\&=\frac{1}{l}\sum_{i=1}^{l}\left(y_i^2-2y_i\frac{1}{l}\sum_{j=1}^{l}y_j+\frac{1}{l^2}\left(\sum_{j=1}^{l}y_j\right)^2\right)\\&=\frac{1}{l}\left(\sum_{i=1}^{l}y_i^2-\frac{2}{l}\left(\sum_{i=1}^{l}y_i\right)^2+\frac{1}{l}\left(\sum_{j=1}^{l}y_j\right)^2\right)\\&=\frac{1}{l}\left(\sum_{i=1}^{l}y_i^2-\frac{1}{l}\left(\sum_{i=1}^{l}y_i\right)^2\right)\end{aligned}$$

根据样本集的回归误差，我们同样可以构造出分裂的回归误差。分裂的目标是最大程度的减小回归误差，因此把分裂的误差指标定义为分裂之前的回归误差减去分裂之后左右子树的回归误差

$$E=E(D)-\frac{N_L}{N}E\left(D_L\right)-\frac{N_R}{N}E\left(D_R\right)$$

将误差的计算公式代入上式，可以得到：

$$E=\frac{1}{N}\left(\sum_{i=1}^{N}y_i^2-\frac{1}{N}\left(\sum_{i=1}^{N}y_i\right)^2\right)-\frac{N_L}{N}\left(\frac{1}{N_L}\left(\sum_{i=1}^{N_L}y_i^2-\frac{1}{N_L}\left(\sum_{i=1}^{N_L}y_i\right)^2\right)\right)-$$

$$\frac{N_R}{N}\left(\frac{1}{N_R}\left(\sum_{i=1}^{N_R}y_i^2-\frac{1}{N_R}\left(\sum_{i=1}^{N_R}y_i\right)^2\right)\right)$$

$$=-\frac{1}{N^2}\left(\sum_{i=1}^{N}y_i\right)^2+\frac{1}{N}\left(\frac{1}{N_L}\left(\sum_{i=1}^{N_L}y_i\right)^2+\frac{1}{N_R}\left(\sum_{i=1}^{N_R}y_i\right)^2\right)$$

由于 $N$ 和 $-\frac{1}{N^2}\left(\sum_{i=1}^{N}y_i\right)^2$ 是常数，要让上式最大化等价于让下式最大化

$$E=\frac{1}{N_L}\left(\sum_{i=1}^{N_L}y_i\right)^2+\frac{1}{N_R}\left(\sum_{i=1}^{N_R}y_i\right)^2$$

寻找最佳分裂时要计算上面的值，让该值最大化的分裂就是最佳分裂。回归树对类别型特征的处理和分类树类似，只是 $E$ 值的计算公式不同，其他的过程相同。

### 5.3.3 叶子节点值的设定

如果不能继续分裂，则将该节点设置为叶子节点。对于分类树，将叶子节点的值设置成本节点的训练样本集中出现概率最大的那个类；对于回归树，则设置为本节点训练样本标签值的均值。

### 5.3.4 属性缺失问题

在某些情况下样本特征向量中一些分量没有值，这称为属性缺失。例如晚上我们无法观察到物体的颜色值，颜色属性就缺失了。在决策树的训练过程中，寻找最佳分裂时如果某一个属性上有些样本有属性缺失，可以把这些缺失该属性的样本剔除掉，然后照常训练，这是最简单的做法。

此外还可以使用替代分裂规则。对于每个决策树节点除了计算出一个最佳分裂规则作为主分裂规则，还会生成一个或者多个替代分裂规则作为备选。在预测时如果主分裂规则对应的特征出现缺失，则使用替代分裂规则进行判定。

现在的关键问题是怎样生成替代分裂规则。其目标是对训练样本的分裂结果要和主分裂尽可能接近，即被主分裂分到左边的样本要尽量被替代分裂分到左边；被主分裂分到右边的样本要尽量被替代分裂分到右边。主分裂和替代分裂对所有训练样本的分裂结果有 4 种情况，分别是：

$$\text{LL},\text{LR},\text{RL},\text{RR}$$

LL 表示被主分裂、替代分裂都分到了左子树的样本数。LR 表示被主分裂分到了左子树，被替代分裂分到了右子树的样本数。RL 表示被主分裂分到了右子树，被替代分裂分到了左子树的样本数。RR 表示被主分裂和替代分裂都分到了右子树的样本数。

因此 LL+RR 是主分裂和替代分裂的结果一致的样本数，LR+RL 是主分裂和替代分裂的结果不一致的样本数。由于可以将左右子树反过来，因此给定一个特征分量，在寻找替代分裂的分裂阈值时要让 LL+RR 或者 LR+RL 最大化，最后取它们的最大值：

$$\max\left(\mathrm{LL}+\mathrm{RR},\mathrm{LR}+\mathrm{RL}\right)$$

该值对应的分裂阈值为替代分裂的分裂阈值。对于除最佳分裂所用特征之外的其他所有特征，都找出该特征的最佳分裂和上面的值。最后取该值最大的那个特征和分裂阈值作为替代分裂规则。

对单个特征寻找替代分裂阈值的处理流程如下：

1.对于每个特征将 $l$ 个训练样本按照该特征的值从小到大排序，假设排序后的值为：

$$x_1,...,x_l$$

接下来从 $x_1$ 开始，依次用每个 $x_i$ 作为阈值，将样本分成左右两部分，同时用主分裂对这些样本进行预测，得到 $\mathrm{LL},\mathrm{LR},\mathrm{RL},\mathrm{RR}$ 的值。

2.将 $\mathrm{LL}+\mathrm{RR},\mathrm{LR}+\mathrm{RL}$ 分别与它们的最大值比较，如果大于最大值，则更新最大值。

3 返回 $\max\left(\mathrm{LL}+\mathrm{RR},\mathrm{LR}+\mathrm{RL}\right)$ 对于的分裂阈值。

这一过程类似于寻找最佳分裂，但采用了不同的评价指标。得到每个特征分量的最佳分裂阈值以及 $\max\left(\mathrm{LL}+\mathrm{RR},\mathrm{LR}+\mathrm{RL}\right)$ 之后，比较各个特征的 $\max\left(\mathrm{LL}+\mathrm{RR},\mathrm{LR}+\mathrm{RL}\right)$ 值，取该值最大的那个特征作为替代分裂特征，对应的阈值作为替代分裂阈值。

### 5.3.5 剪枝算法

如果决策树的结构过于复杂，可能会导致过拟合问题。此时需要对树进行剪枝，消掉某些节点让它变得更简单。剪枝的关键问题是确定减掉哪些树节点以及减掉它们之后如何进行节点合并。决策树的剪枝算法可以分为两类，分别称为预剪枝和后剪枝。前者在树的训练过程中通过停止分裂对树的规模进行限制；后者先构造出一棵完整的树，然后通过某种规则消除掉部分节点，用叶子节点替代。

预剪枝可以通过限定树的高度，节点的训练样本数，分裂所带来的纯度提升的最小值来来实现，具体做法在前面已经讲述，在源代码分析中会介绍实现细节。后剪枝的典型实现有降低错误剪枝（Reduced-Error Pruning，简称 REP）、悲观错误剪枝（Pesimistic-Error Pruning，简称 PEP）、代价-复杂度剪枝（Cost-Complexity Pruning，简称 CCP）[5]等方案。分类与回归树采用的是代价-复杂度剪枝算法，下面重点介绍它的原理。

代价是指剪枝后导致的错误率的变化值，复杂度是指决策树的规模。训练出一棵决策树之后，剪枝算法首先计算该决策树每个非叶子节点的 $\alpha$ 值，它是代价与复杂度的比值。该值定义为：

$$\alpha=\frac{E\left(n\right)-E\left(n_t\right)}{\left|n_t\right|-1}$$

其中 $E\left(n\right)$ 是节点 $n$ 的错误率，$E\left(n_t\right)$ 是以节点 $n$ 为根的子树的错误率，$\left|n_t\right|$ 为子树的叶子节点数，即复杂度。$\alpha$ 值是用树的复杂度归一化之后的错误率增加值，即将整个子树剪掉之后用一个叶子节点替代，相对于原来的子树错误率的增加值。该值越小，剪枝之后树的分类效果和剪枝之前越接近。上面的定义依赖于节点的错误率指标，下面对分类问题和回归问题介绍它的计算公式。对于分类问题，错误率定义为

$$E(n)=\frac{N-\max(N_i)}{N}$$

其中 $N$ 是节点的总样本数，$N_i$ 是第 $i$ 类样本数，这就是之前定义的误分类指标。对于回归问题，错误率为节点样本集的均方误差

$$E(n)=\frac{1}{N}\left(\sum_i(y_i^2)-\frac{1}{N}\left(\sum_i y_i\right)^2\right)$$

子树的错误率为树的所叶子节点错误率之和。计算出 $\alpha$ 值之后，剪掉该值最小的节点得到剪枝后的树，然后重复这种操作直到剩下根节点，由此得到一个决策树序列

$$T_0,...,T_m$$

其中 $T_0$ 是初始训练得到的决策树，$T_{i+1}$ 在 $T_i$ 的基础上剪枝得到的，即剪掉 $T_i$ 中 $\alpha$ 值最小的那个节点为根的子树用一个叶子节点替代后得到的树。

整个剪枝算法分为两步完成。

第一步先训练出 $T_0$，然后用上面的方法逐步剪掉树的所有非叶子节点，直到只剩下根节点，得到剪枝后的树序列。这一步的误差计算采用的是训练样本集。

第二步根据真实误差值从上面的树序列中挑选出一棵树作为剪枝后的结果。这可以通过交叉验证实现，用交叉验证的测试集对上一步得到的树序列的每一棵树进行测试，得到这些树的错误率，然后根据错误率选择最佳的树作为剪枝后的结果。

### 5.3.6 训练算法的流程

下面给出决策树完整的训练算法。算法的输入为训练样本集，输出为训练得到的树。训练算法 TrainDecisionTree 的流程为

```
TrainDecisionTree(D)
if (样本集无法再分裂 或 达到最大树深度 或 D 的样本数小于指定阈值)
    leafNode =  CalcLeafValue(D); // 无法再分裂，设置为叶子节点，计算其值
    return leafNode;// 返回创建的叶子节点
else
    (split, D1, D2) = FindBestSplit(D); // 寻找最佳分裂 split，将训练集 D 分为 D1 和 D2
    node = CreteTreeNode(); // 创建当前节点
    node->split = split; // 设置节点的分裂规则
    FinSurrogateSplit(D); //寻找替代分裂，加入到节点的分裂规则列表
    node->leftChild = TrainDecisionTree(D1); // 递归训练左子树
    node->rightChild = TrainDecisionTree(D2); // 递归训练右子树
    return node; // 泛化训练的树节点的
end if
```

如果需要做后剪枝处理，训练结束之后还要调用剪枝函数。

### 5.3.7 计算变量的重要性

决策树可以输出特征分量的重要性，即每个特征分量对分类或回归的作用大小。计算方

法为对每个特征分量在整个决策树中的分裂质量累加求和。对于分类树，分裂质量是 Gini 纯度值，即 5.3.2 节中定义的 $G$ 值；对于回归树，分裂质量是每次分裂时回归误差的下降值，即 5.3.2 节中定义的 $E$ 值。假设第 $i$ 个特征分量的分裂质量之和为 $q_i$，然后对所有特征分量求和后分裂质量进行归一化

$$\frac{q_i}{\sum_{i=1}^{n} q_i}$$

这个值即为该特征分量的重要性。其中 $n$ 为特征向量的维数。显然该值越大变量越重要。统计所有节点的分裂质量需要对树进行遍历，可以使用任何一种遍历算法。这样做的依据是，如果一个变量被选来做分裂则说明它对分类或者回归很重要，如果它做分裂时的分裂质量很大，说明其对分类或者回归的贡献很大。

## 5.4 实验程序

下面通过实验程序介绍决策树的使用。程序基于 sklearn，使用 iris 数据集。具体做法与第 4.4 节相同。训练决策树时需要指定一些参数，包括分裂质量的度量指标，决策树的最大深度，允许分裂的最小训练样本数等。这里使用了默认的参数。程序源代码可以通过左侧二维码获取。

程序的运行结果如下图 5.5 所示。

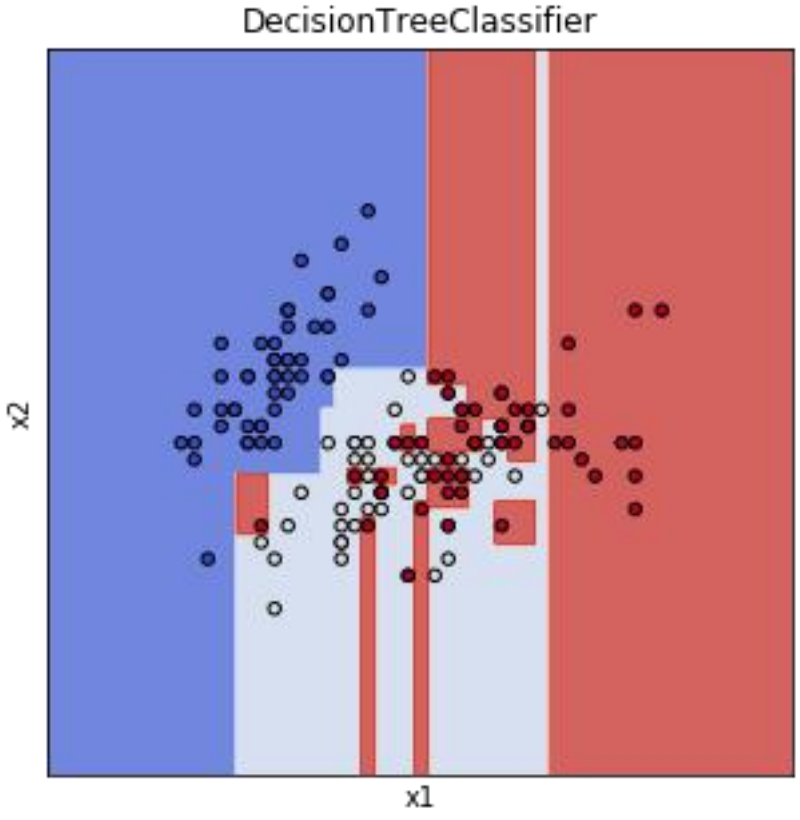


图 5.5　决策树对 iris 数据集的分类结果

上图 5.5 中的分类决策线为直线段，这也证明了决策树是分段线性函数。

## 5.5 应用

决策树具有实现简单，计算量小的优点，并具有很强的可解释性。训练得到的树模型符合人的直观思维，能够可视化的显示出来，因此便于理解，这对某些数据的分析非常重要。它被成功应用于经济和管理数据分析、疾病诊断、模式识别等各类问题。除了单独使用之外，决策树还作为弱分类器用于随机森林和 AdaBoost 等集成学习算法，在第 12 章和第 13 章中将会详细介绍。

## 参 考 文 献


[1] J. Ross Quinlan. Induction of decision trees. Machine Learning, 1(1): 81-106, 1986.

[2] J. Ross Quinlan. Learning efficient classification procedures and their application to chess end games. 1993.

[3] J. Ross Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco, CA, 1993.

[4] Breiman, L., Friedman, J. Olshen, R. and Stone C. Classification and Regression Trees, Wadsworth, 1984.

[5] Eibe Frank. Pruning Decision Trees and Lists. 2000.