第5章　其他典型分类方法

5.1　近　邻　法

近邻法是由 Cover 和 Hart 于 1968 年提出的，是最重要的非参数分类方法之一。已知贝叶斯分类器在特殊条件下可以简化为最小距离分类器，即通过计算待识别样本到每类样本均值的距离，将待识别样本归为具有最小距离的那一类。对最小距离分类器进行拓展，将训练样本集划分为若干子类，从每个子类中选取一个代表点，将待识别样本归为最近代表点所属的类别。近邻法将训练样本集中的每个样本都作为代表点，实质上是一种分段线性分类器。

5.1.1　最近邻法

最近邻分类器是最小距离分类器的一种极端情况，以全部训练样本作为代表点，计算待识别样本与所有训练样本的距离，并将待识别样本归为最近邻的训练样本所属类别。

假定样本集有 m 个类别 $\omega_1,\omega_2,\cdots,\omega_m$，每个类别中有 $N_i(i=1,2,\cdots,m)$ 个样本，以每个训练样本作为一个子类，计算待识别样本 $\boldsymbol{x}$ 与每一个子类的距离，则将样本归为 ω_i 类的判别函数为

$$g_i(\boldsymbol{x})=\min_i \|\boldsymbol{x}-\boldsymbol{x}_i^k\|,\quad k=1,2,\cdots,N_i \tag{5-1}$$

其中，$\boldsymbol{x}_i^k$ 表示第 i 类的第 k 个样本。此时，判别准则为

$$g_l(\boldsymbol{x})=\min_{i=1,2,\cdots,m} g_i(\boldsymbol{x}),\text{则 } \boldsymbol{x}\in\omega_l \tag{5-2}$$

当训练样本数量无限增多时，待识别样本 $\boldsymbol{x}$ 的最近邻样本在极限意义上讲就是 $\boldsymbol{x}$ 本身。对于两类别的分类问题，假设待识别样本 $\boldsymbol{x}$ 属于两类的后验概率分别为 $P(\omega_1\mid\boldsymbol{x})$ 和 $P(\omega_2\mid\boldsymbol{x})$，则当待识别样本与它的最近邻样本都属于同一类时，才能决策正确，因此正确分类率为 $\sum_{i=1}^{C}P^2(\omega_i\mid\boldsymbol{x})$。所以，当训练样本数量无限多时，待识别样本 x 分类错误的概率为

$$\lim_{N\to\infty}P_N(e\mid X)=1-\sum_{i=1}^{c}P^2(\omega_i\mid\boldsymbol{x}) \tag{5-3}$$

在该条件下的平均错误率为

$$\begin{aligned}P&=\lim_{N\to\infty}P_N(e)\\&=\lim_{N\to\infty}\int P_N(e\mid X)P(X)\mathrm{d}X\end{aligned}$$

$$=\int \lim_{N\to\infty} P_N(e \mid X)P(X)\mathrm{d}X$$

$$=\int\left[1-\sum^{c} P^2(\omega_i \mid \boldsymbol{x})\right]P(X)\mathrm{d}X \tag{5-4}$$

P 也被称为渐进平均错误率。

贝叶斯错误率为

$$P^* =\int P^*(e \mid X)P(X)\mathrm{d}X \tag{5-5}$$

其中，$P^*(e \mid X)=1-P^*(\omega_m \mid X)$，$P^*(\omega_m \mid X)=\max_i[P(\omega_i \mid X)]$

最近邻法的错误率虽然高于贝叶斯错误率，但是当样本数目无限时，最近邻法的错误率不会超过贝叶斯错误率的两倍。令最近邻法分类器的错误率为 P，贝叶斯分类器的错误率为 P^*，那么它们之间存在如下关系：

$$P^* \leqslant P \leqslant P^*\left(2-\frac{m}{m-1}P^*\right) \tag{5-6}$$

其中，m 为类别数。一般情况下，P^* 很小，因此也可粗略地表示为

$$P^* \leqslant P \leqslant 2P^* \tag{5-7}$$

5.1.2 k-近邻法

尽管最近邻样本和待识别样本在距离意义下是最相似的，但是如果最近邻样本和待识别样本不属于同一类别，则会直接导致分类错误。因此，最近邻法的缺点在于受随机噪声影响较大，尤其是在两类的交叠区域。

k-近邻法(k-Nearest Neighbor)是最近邻法的扩展。k-近邻法的基本规则是：在所有 N 个样本中找到待识别样本的 k 个最近邻者，设 k 个样本中属于第 i 类的样本有 k_i 个，即

$$k=k_1+k_2+\cdots+k_m \tag{5-8}$$

则定义判别函数为

$$g_i(\boldsymbol{x})=k_i, \quad i=1,2,\cdots,m \tag{5-9}$$

此时，判别准则为

$$若 g_j(\boldsymbol{x})=\max_i k_i，则 \boldsymbol{x}\in\omega_j \tag{5-10}$$

我们称这种方法为 k-近邻法，相应的分类器称为 k-近邻分类器。k-近邻法中的 k 一般采用奇数，类似于投票表决，避免因两种票数相等而难以决策。k-近邻法可以理解为，从样本点 $\boldsymbol{x}$ 开始生长，不断扩大区域，直到包含 k 个训练样本为止，并且把待识别样本点 $\boldsymbol{x}$ 的类别归为最近的 k 个训练样本点中出现频率最高的样本对应的类别。

k-近邻法错误率的上下界同最近邻法一样，均在一倍到两倍贝叶斯分类器的错误率范围内。在 $k\to\infty$ 的条件下，k-近邻法的错误率要低于最近邻法，同时趋近于贝叶斯错误率。

例 5.1 有 7 个二维样本向量：$\boldsymbol{x}_1=\begin{pmatrix}1\\0\end{pmatrix}$，$\boldsymbol{x}_2=\begin{pmatrix}0\\1\end{pmatrix}$，$\boldsymbol{x}_3=\begin{pmatrix}0\\-1\end{pmatrix}$，$\boldsymbol{x}_4=\begin{pmatrix}0\\0\end{pmatrix}$，$\boldsymbol{x}_5=\begin{pmatrix}0\\2\end{pmatrix}$，$\boldsymbol{x}_6=\begin{pmatrix}0\\-2\end{pmatrix}$，$\boldsymbol{x}_7=\begin{pmatrix}-2\\0\end{pmatrix}$，假定前 3 个为 ω_1 类，后 4 个为 ω_2 类。试分别根据最近邻法和 k-近邻法

($k=3$)判断 $\boldsymbol{x}=\begin{pmatrix}1\\2\end{pmatrix}$ 属于哪一类别。

解：首先计算样本 $\boldsymbol{x}=\begin{pmatrix}1\\2\end{pmatrix}$ 与 7 个样本之间的距离：

$$d_1=\sqrt{(1-1)^2+(2-0)^2}=2 \qquad d_2=\sqrt{(1-0)^2+(2-1)^2}=\sqrt{2}$$

$$d_3=\sqrt{(1-0)^2+(2+1)^2}=\sqrt{10} \qquad d_4=\sqrt{(1-0)^2+(2-0)^2}=\sqrt{5}$$

$$d_5=\sqrt{(1-0)^2+(2-2)^2}=1 \qquad d_6=\sqrt{(1-0)^2+(2+2)^2}=\sqrt{17}$$

$$d_7=\sqrt{(1+2)^2+(2-0)^2}=\sqrt{13}$$

(1) 最近邻法。

对于 ω_1 类：$g_1(\boldsymbol{x})=\min\{2,\sqrt{2},\sqrt{10}\}=\sqrt{2}$。

对于 ω_2 类：$g_2(\boldsymbol{x})=\min\{\sqrt{5},1,\sqrt{17},\sqrt{13}\}=1$。

根据最近邻法，因为 $g_2(\boldsymbol{x})<g_1(\boldsymbol{x})$，所以 $\boldsymbol{x}=\begin{pmatrix}1\\2\end{pmatrix}$ 属于 ω_2 类。

(2) k-近邻法($k=3$)。

最近的 3 个距离为：$1,\sqrt{2},2$。

3 个近邻分别为：$\boldsymbol{x}_5=(0,2)^{\mathrm{T}},\boldsymbol{x}_2=(0,1)^{\mathrm{T}},\boldsymbol{x}_1=(1,0)^{\mathrm{T}}$。

3 个近邻中有两个属于 ω_1 类，有一个属于 ω_2 类，所以 $\boldsymbol{x}$ 属于 ω_1 类。

5.1.3　改进的近邻法

近邻法的改进原理大致可分为以下两种：一种是对样本集进行组织与整理，分群分层，尽可能将计算压缩到在接近测试样本邻域的小范围内，避免盲目地与训练样本集中每个样本进行距离计算，例如快速搜索近邻法。另一种是在原有样本集中挑选出对分类计算有效的样本，使样本总数合理地减少，以同时达到既减少计算量，又减少存储量的双重效果，例如剪辑近邻法和压缩近邻法。

快速搜索近邻法的基本思想是：将样本集按近邻关系分解成组，给出每组的质心所在，以及组内样本到质心的最大距离。这些组又可形成层次结构，将组分为子组，类似树的结构。因此待识别样本可将搜索近邻的范围从某一大组逐渐深入到其中的子组，直到树的叶节点所代表的组，确定其相邻关系。这种方法可以有效减少计算量，但不能减少存储量。

令 p 代表树中的一个节点，K_p 是 p 对应的一个样本子集，N_p 是 K_p 对应的一个样本子集，M_p 代表 K_p 中的样本均值，r_p 代表 K_p 中任一样本到 M_p 的最大距离。快速搜索近邻法包含两个规则，如下。

规则一：如果待识别样本 x 满足 $D(x,M_p)>B+r_p$，则 x 的最近邻不在 K_p 中。

规则二：如果待识别样本 x 满足 $D(x,M_p)>B+D(x_i,M_p)$，则 x_i 不是 x 的最近邻。

快速搜索近邻法的搜索过程如下。

(1)(初始化)：$B=\infty$，$L=0$(当前水平)，$p=0$(当前节点)。

(2)(当前节点展开)：将当前节点的所有直接后继节点放入一个目录表(活动表)中，对它们计算并存储 $D(x,M_p)$。

(3)(规则一检验):对活动表中的每个节点,采用规则一将不符合条件的节点从活动表中去掉。

(4)(回溯):如果活动表为空(当前节点的子节点被全部剪掉),则回溯到上一级,即 $L=L-1$;如果 $L=0$,终止算法,如果活动表中存在一个以上的节点,转到步骤(5)。

(5)(选择最近节点):在活动表中选择使 $D(x,M_p)$ 最小的节点 P^* 作为当前节点,若 L 为最终水平,则转到步骤(6),否则置 $L=L+1$,转到步骤(2)。

(6)(规则二检验):对当前节点 P^* 中的每个点 x_i,用规则二决定是否计算 $D(x,x_i)$,若 $D(x,x_i)<B$,那么最近邻为 x_i,$B=D(x,x_i)$,检验完 P^* 中的所有 x_i 后转到步骤(3)。

剪辑近邻法的基本思想是:当不同类别的样本在分布上有交叠部分时,分类的错误率主要来自处于交叠区中的样本。由于交叠区域中不同类别的样本彼此穿插,会导致用近邻法分类出错,因此如果能将不同类别交界处的样本以适当方式筛选,可以实现既减少样本数又提高正确识别率的双重目的。为实现上述目的,可以利用现有样本集对其自身进行剪辑,基本思路是:考查样本是否为可能的误识别样本,若是,则从样本集中去掉。剪辑近邻法最终去掉两类边界附近的样本,能在一定程度上减少样本数量,因此能够降低错误率。剪辑的过程如下:首先将样本集分成两个互相独立的子集,即考试集 K^T 和参考集 K^R。对考试集 K^T 中的每一个样本 x_i,在参考集 K^R 中找到最近邻样本 y_i。如果 y_i 和 x_i 不属于同一类别,则将 x_i 从考试集 K^T 中删除。重复以上样本剪辑过程,最终可以得到一个剪辑样本集 K^{TE},以取代原样本集,对待识别样本进行最近邻分类。

剪辑近邻法的结果只是去掉了两类边界附近的样本,而靠近两类中心的样本几乎没有被去掉。压缩近邻法的基本思想是在样本剪辑的基础上再去掉一部分靠近两类中心的样本,这有助于进一步缩短计算时间和降低存储要求。压缩近邻法中定义了两个存储器 A 和 B,其中 A 用来存放即将生成的样本集,B 用来存放原样本集。压缩近邻法的步骤如下。

(1) 随机挑选一个样本,放在存储器 A 中,其他样本放在存储器 B 中。

(2) 用当前存储器 A 中的样本按最近邻法对存储器 B 中的样本进行分类,假如分类正确,该样本放回存储器 B;否则放入存储器 A。

(3) 重复上述过程,直到在执行中没有一个样本从存储器 B 转到存储器 A,或者存储器 B 为空为止。

5.2 支持向量机

支持向量机(Support Vector Machine,SVM)是由 Vapnik 等于 1995 年提出的一种分类器设计方法,已广泛应用在许多模式识别系统中。SVM 是一种基于统计学习理论的方法,建立在统计学习理论的 VC 维理论和结构风险最小化原则之上。下面以两分类问题为例介绍,针对两分类问题,SVM 在高维空间中寻找一个超平面作为两类的分界面,以保证分类的错误率最小。少量与分界面比较接近的训练样本称为支持向量,它们决定了分类器的推广能力。

5.2.1　线性可分的情况

SVM 最初是从线性可分的情况发展而来的。如图 5-1 所示，圆点和方点各代表一类样本，H 为分界线，H_1 和 H_2 分别为过两类中距离分界线最近的样本且与分界线平行的直线。H_1 和 H_2 之间的距离称为分类间隔，处在隔离带边缘上的样本称为支持向量。最优分类线就是要求分类线不但能将两类样本正确分类，而且使分类间隔最大。推广到高维空间，最优分类线就变为了最优分类面。

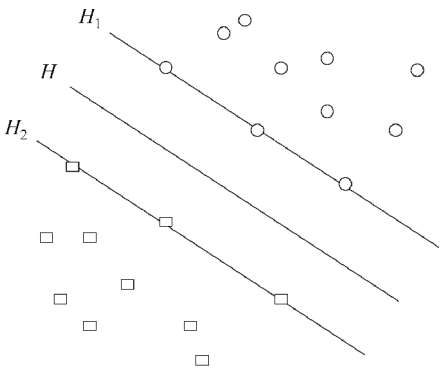


图 5-1　线性可分情况下的 SVM 最优分类

假设存在训练样本 $(\boldsymbol{x}_i, y_i), i=1,2,\cdots,N$，$\boldsymbol{x}_i \in \boldsymbol{R}^n$，$y_i \in \{-1,+1\}$，$y_i=+1$ 代表圆点，$y_i=-1$ 代表方点。在线性可分的情况下，有一个超平面将这两类样本完全分开。超平面可以由一个线性判别函数表示，线性判别函数是由 $\boldsymbol{x}$ 的各个分量通过线性组合得到的函数，它的表达式为

$$g(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b \tag{5-11}$$

$g(\boldsymbol{x})=0$ 定义了一个决策面，其中 $\boldsymbol{w}$ 是超平面的法向量，位置由阈值 b 的大小确定，对于目前的两类情况，给定样本 $\boldsymbol{x}$，判别规则如下：

若 $g(\boldsymbol{x})>0$，判定 $\boldsymbol{x}$ 属于圆点类；

若 $g(\boldsymbol{x})<0$，判定 $\boldsymbol{x}$ 属于方点类；

若 $g(\boldsymbol{x})=0$，任意对 $\boldsymbol{x}$ 分类或者拒判。

SVM 的目标可以表达为：找到一个超平面，使得它能够尽可能多地将两类数据点正确地分开，同时使分开的两类数据点距离分类面最远。

n 维空间中的判别函数为 $g(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b$，分类面方程为 $H:\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=0$。寻找合适的 $\boldsymbol{w}$ 和 b，使其能够达到 SVM 的目标。

$$H_1:\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=k_1 \tag{5-12}$$

$$H_2:\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=k_2 \tag{5-13}$$

令 $k=\dfrac{k_1-k_2}{2}$，并代入 H_1、H_2 中，可以得到：

$$H_1:\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b-k_1+k=k;\left(b-k_1+k=b-\frac{k_1+k_2}{2}=\widetilde{b}\right) \tag{5-14}$$

$$H_2:\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b-k_2+k=-k;\left(b-k_2+k=b-\frac{k_1+k_2}{2}=\widetilde{b}\right) \tag{5-15}$$

化简可得

$$H_1:\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+\widetilde{b}=k \tag{5-16}$$

$$H_2:\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+\widetilde{b}=-k \tag{5-17}$$

将式(5-12)和式(5-13)进行归一化处理，可以得到：

$$H_1:\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=1 \tag{5-18}$$

$$H_2: \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=-1 \tag{5-19}$$

通过将判别函数归一化，使得两类所有样本都满足 $|g(\boldsymbol{x})|\geqslant 1$，离分类面 H 最近的样本满足 $|g(\boldsymbol{x})|=1$。因此判别函数满足如下条件：

$$g(\boldsymbol{x}_i)=\begin{cases}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b\geqslant 1, & y_i=1\\ \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b\leqslant -1, & y_i=-1\end{cases} \tag{5-20}$$

综合两个条件，可以得到 $y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)\geqslant 1, i=1,2,\cdots,N$。

任意一点 $\boldsymbol{x}$ 到超平面的距离 r 可以表示为：$r=\dfrac{g(x)}{\|\boldsymbol{w}\|}$，原点到超平面的距离可以表示为：$\dfrac{b}{\|\boldsymbol{w}\|}$。最优化超平面是由最大化间隔 ρ 给出，在满足归一化后的条件下，ρ 可以表示为

$$\rho=\min_{x_i:y_i=-1}\frac{|\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b|}{\|\boldsymbol{w}\|}+\min_{x_i:y_i=1}\frac{|\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b|}{\|\boldsymbol{w}\|}=\frac{2}{\|\boldsymbol{w}\|} \tag{5-21}$$

因此分类间隔等于 $\dfrac{2}{\|\boldsymbol{w}\|}$，使 $\dfrac{2}{\|\boldsymbol{w}\|}$ 最大等价于使 $\|\boldsymbol{w}\|^2$ 最小。综上可以得到，满足 $y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)\geqslant 1, i=1,2,\cdots,N$ 以及 $\|\boldsymbol{w}\|^2$ 最小的分界面称为最优分界面，H_1、H_2 上的训练样本点称为支持向量。

统计学习理论指出：在 n 维空间中，设样本分布在一个半径为 R 的超球形范围内，则满足条件 $\|\boldsymbol{w}\|\leqslant A$ 的正则超平面构成的指示函数集为

$$f(\boldsymbol{x},\boldsymbol{w},b)=\mathrm{sgn}\{\langle \boldsymbol{w},\boldsymbol{x}\rangle+b\} \tag{5-22}$$

其中 sgn()为符号函数，其 VC 维 h 满足式(5-18)表明的界：

$$h\leqslant \min(|R^2A^2|,N)+1 \tag{5-23}$$

因此，使 $\|\boldsymbol{w}\|^2$ 最小就变成了求下面的函数解：

$$V(\boldsymbol{w},b)=\frac{1}{2}\langle \boldsymbol{w},\boldsymbol{w}\rangle \tag{5-24}$$

求解 SVM 的问题转换为对变量 $\boldsymbol{w}$ 和 b 的最优化问题：

$$\min_{\boldsymbol{w},b}\frac{1}{2}\|\boldsymbol{w}\|^2 \tag{5-25}$$

$$y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)\geqslant 1,\quad i=1,2,\cdots,N \tag{5-26}$$

可以看出，最优化问题是约束条件为不等式的条件极值问题，可以引用扩展的拉格朗日乘子理论求解，可构造拉格朗日函数：

$$L(\boldsymbol{w},b,\alpha)=\frac{1}{2}\|\boldsymbol{w}\|^2-\sum_{i=1}^{N}\alpha_i[y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)-1] \tag{5-27}$$

根据 K-T 条件以及极值存在条件，可以得到：

$$\frac{\partial L(\boldsymbol{w},b,\alpha)}{\partial \boldsymbol{w}}=\boldsymbol{w}-\sum_{i=1}^{N}\alpha_i y_i x_i=0 \tag{5-28}$$

$$\frac{\partial L(\boldsymbol{w},b,\alpha)}{\partial b}=\sum_{i=1}^{N}\alpha_i y_i=0 \tag{5-29}$$

$$\alpha_i=[y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)-1]=0,\quad \alpha_i\geqslant 0, i=1,2,\cdots,N \tag{5-30}$$

结合式(5-21)的条件，通过式(5-25)表明：只有满足 $y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)-1=0$ 条件的点，其拉格朗日乘子才可能不为 0；而对满足 $y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)-1>0$ 的样本数据来说，其拉格朗日乘子必须为 0。根据式(5-23)可以得到：

$$\boldsymbol{w}^{*}=\sum_{i}\alpha_i^{*}\,y_i\boldsymbol{x}_i \tag{5-31}$$

显然，只有部分样本数据的 α_i 不为 0，而线性分界面的权向量 $\boldsymbol{w}$ 是这些 α_i 不为 0 的样本数据的线性组合，因而 α_i 不为 0 的样本数据也被称为支持向量。

为了求取最佳的 α_i，拉格朗日引入一种对偶函数，对偶函数通过对 $L(w,b,\alpha)$ 函数求其对 $\boldsymbol{w}$ 及 b 的偏微分，并置零，再代回到拉格朗日函数中，得到：

$$L(\alpha)=\sum_{i=1}^{N}\alpha_i-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j\,y_i\,y_j<x_i\cdot x_j> \tag{5-32}$$

此时变为求 $-L(\alpha)$ 的最小值问题：

$$\min\left(\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j\,y_i\,y_j<x_i\cdot x_j>-\sum_{i=1}^{N}\alpha_i\right) \tag{5-33}$$

$$\text{s.t.}\sum_{i=1}^{N}\alpha_i\,y_i=0,\quad \alpha_i\geqslant 0,i=1,2,\cdots,N \tag{5-34}$$

因此，最后求得最优权向量 $\boldsymbol{w}^{*}=\sum_{i}\alpha_i^{*}\,y_i\boldsymbol{x}_i$。可以看出，最优权向量为训练样本中的支持向量的线性组合。最优阈值 b^{*} 的确定有两种求取方法：

第一种，选择 α_i^{*} 的一个分量 $\alpha_j^{*}>0$：

$$b^{*}=y_j-\sum_{i=1}^{N}y_i\alpha_i^{*}<x_i\cdot x_j> \tag{5-35}$$

第二种：

$$b^{*}=-\frac{\max\limits_{x_i;y_i=-1}\boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x}_i+\min\limits_{x_i;y_i=+1}\boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x}_i}{2} \tag{5-36}$$

由此求得决策函数 $g(\boldsymbol{x})=\boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x}+b^{*}$。

在实际情况中，并非所有的点可以用一条直线划分，如果继续用直线划分，必然会出现错分点。因此，放宽要求，希望错分的程度尽可能小，可以在条件中增加松弛项 $\xi_i\geqslant 0$，约束条件放宽为 $y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)+\xi_i\geqslant 1$。此时目标函数变为

$$\Phi(\boldsymbol{w},\xi)=\frac{1}{2}\|\boldsymbol{w}\|^{2}+C\sum_{i=1}^{N}\xi_i \tag{5-37}$$

其中，C 为可调参数，表示对错误的惩罚程度，C 越大，惩罚越重。因此非线性问题可以描述为

$$\min_{\boldsymbol{w},b,\xi}\frac{1}{2}\|\boldsymbol{w}\|^{2}+C\sum_{i=1}^{N}\xi_i \tag{5-38}$$

$$\text{s.t. }y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)+\xi_i\geqslant 1,\quad i=1,2,\cdots,N \tag{5-39}$$

$$\xi_i\geqslant 0,\quad i=1,2,\cdots,N \tag{5-40}$$

与前述类似，引入其对偶问题：

$$\min_{\alpha}\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}y_i\,y_j\,\alpha_i\alpha_j<x_i\cdot x_j>-\sum_{i=1}^{N}\alpha_i \tag{5-41}$$

$$\text{s.t.}\ \sum_{i=0}^{l} y_i \alpha_i = 0 \tag{5-42}$$

$$0 \leqslant \alpha_i \leqslant C, \quad i = 1,2,\cdots,N \tag{5-43}$$

求解上述最优化问题的最优解 $\boldsymbol{w}^*$、b^*，则决策函数为 $g(\boldsymbol{x}) = \boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x} + b^*$。

例 5.2 有 4 个二维向量及其对应标签：$\boldsymbol{x}_1 = \begin{pmatrix}0\\0\end{pmatrix}$，$y_1 = +1$，$\boldsymbol{x}_2 = \begin{pmatrix}1\\0\end{pmatrix}$，$y_2 = +1$，$\boldsymbol{x}_3 = \begin{pmatrix}2\\0\end{pmatrix}$，$y_3 = -1$，$\boldsymbol{x}_4 = \begin{pmatrix}0\\2\end{pmatrix}$，$y_4 = -1$，求最优分类面。

解：求解其对偶问题：

$$\min\left(\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j < x_i \cdot x_j > - \sum_{i=1}^{N}\alpha_i\right)$$

$$\text{s.t.}\ \sum_{i=1}^{N}\alpha_i y_i = 0\ \alpha_i \geqslant 0, i = 1,2,3,4$$

代入可得：

$$\max -L(\alpha) = (\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4) - \frac{1}{2}(\alpha_2^2 - 4\alpha_2\alpha_3 + 4\alpha_3^2 + 4\alpha_4^2)$$

$$\text{s.t.}\ \alpha_1 + \alpha_2 - \alpha_3 - \alpha_4 = 0, \quad \alpha_i \geqslant 0, \quad i = 1,2,3,4$$

求得：$\alpha_1 = 0, \alpha_2 = 1, \alpha_3 = 0.75, \alpha_4 = 0.25$。

因此权值 $\boldsymbol{w} = \begin{pmatrix}1\\0\end{pmatrix} - 0.75\begin{pmatrix}2\\0\end{pmatrix} - 0.25\begin{pmatrix}0\\2\end{pmatrix} = \begin{pmatrix}-0.5\\-0.5\end{pmatrix}$

阈值 $b = -\dfrac{\max\limits_{x_i;y_i=-1}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + \min\limits_{x_i;y_i=+1}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i}{2} = -\dfrac{-0.5-1}{2} = \dfrac{3}{4}$

因此最优分类面为：$g(\boldsymbol{x}) = \dfrac{3 - 2x_1 - 2x_2}{4}$

5.2.2 线性不可分情况

对于图 5-2 所示的问题，如果用直线分类，会产生很大的误差，这类问题称为线性不可分问题，这时就必须使用非线性分类学习机进行分类，如图 5-3 所示。对于这类问题，显然不能用超曲面去划分，此时可以通过一个映射，把寻找一个超曲面的问题转换为寻找超平面的问题。

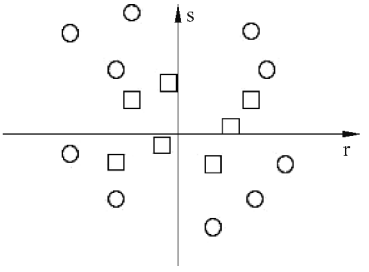


图 5-2 线性不可分问题

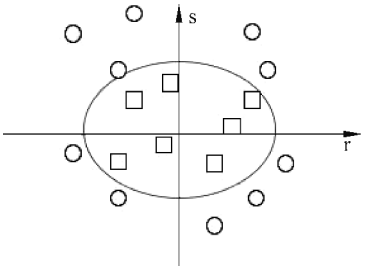


图 5-3 非线性划分

非线性 SVM 的基本思想是：通过非线性变换将非线性问题转换为某个高维空间中的线性问题，在变换空间求最优分类面。一般情况下，新空间维数比原空间维数高。这种映射可表示为：将 x 作变换 $\Phi: R^n \rightarrow H$（H 为某个高维特征空间）

$$x \rightarrow \Phi(x) = (\Phi_1(x), \Phi_2(x), \cdots, \Phi_i(x), \cdots)^{\mathrm{T}} \tag{5-44}$$

其中，$\Phi_i(x)$ 是实函数。通过映射变换后，可以在新空间中建立最优超平面：

$$\langle w, \Phi(x) \rangle + b = 0 \tag{5-45}$$

根据泛函的有关理论，只要一种核函数 $K(x_i, x_j)$ 满足 Mercer 条件，它就对应某一变换空间中的内积，因此可以在这个变换空间中构造线性分类，此时的最优化问题为

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j K(x_i \cdot x_j) - \sum_{i=1}^{N} \alpha_i \tag{5-46}$$

$$\text{s.t.} \sum_{i=0}^{l} y_i \alpha_i = 0 \tag{5-47}$$

$$0 < \alpha_i < C, i = 1, 2, \cdots, N \tag{5-48}$$

其中，

$$K(x_i, x_j) = (\Phi_i(x). \Phi_j(x)) \tag{5-49}$$

通过求解对偶问题来确定最终的决策函数，这样就得到非线性可分支持向量机（标准的支持向量机）算法。

5.3　决　策　树

前面章节中介绍的线性、非线性分类器等分类方法，针对的样本特征都是数值特征。数值特征就是可以被测量的特征，例如人的身高体重、水杯的容量、操场的面积等。带有这种数值特征的样本可以直接输入线性、非线性分类器，得到样本类别。然而在实际生活中，对象的特征有些不是数值特征，如人的性别、国籍、民族，水杯的品牌、功能等，只能比较相同或不相同，无法比较相似性和大小，这类特征叫作名义特征；还有一类特征，有些是数值，如学号、手机号等，有些不是数值，如初中、高中、本科、硕士等学历的级别，都存在顺序，但没有尺度，这类特征叫作序数特征。

对于名义特征，可以采用编码的方式转换成数值特征，比如人的国籍，可以用 001 代表中国，010 代表美国，100 代表法国，但这样增加了特征的维度，而且编码后的特征仍无法作为数值特征使用。对于序数特征，可以根据先验知识转换为数值特征，比如根据学历的高低赋予相应的分数，把分数作为一般的数值特征来处理，但这样做受到人为因素的影响，赋分合理与否对实际分类效果影响较大。

由此可见，虽然有方法可以将非数值特征转换为数值特征，但也存在相应的弊端。直接利用非数值特征对样本分类将有效避免以上问题。决策树方法是一种针对非数值特征的分类方法。在日常生活中，常用决策树的思想做出决策与分类。表 5-1 是某公司入职申请数据。

表 5-1　某公司入职申请数据

编　号	性　别	年　龄	学　历	月　薪	应 届 生	是否录用
1	男	23	本科	10000	是	是
2	男	30	本科	15000	否	否
3	女	25	硕士	17000	是	是
4	男	33	本科	13000	否	否
5	女	36	博士	20000	否	是
6	女	32	本科	18000	否	否
小张	男	22	本科	13000	是	?

小张打算入职该公司，他不知道面试官的决策思路，但他了解编号 1 到 6 员工的信息以及录用结果。虽然没有一例和小张情况相同，但他通过对表 5-1 进行建模得到决策树进行分类，大致得到了自己的申请结果。

5.3.1　基本概念

决策树是常见的分类方法，人们在生活中经常用决策树的思想来做决定。决策树是一种对实例进行分类的树形结构。决策树由一系列节点组成，节点分为内部节点和叶节点，图 5-4 为小张画出的决策树。方形的节点为内部节点，椭圆的节点为叶节点，顶部的内部节点称为根节点(“树”是倒置的，即根在顶部，叶在底部)。每一个内部节点代表一个属性和相应的决策规则，内部节点下面的分支表示不同的判断结果，如果分支后面是叶节点，说明已经能够得到该样例的分类，如小张是否被录用；如果分支后面是内部节点，则需要根据该样例的其他属性做出判断，直到遇到叶节点。如果决策树中每个内部节点下都只有两条分支，那么可以被称为二叉树，否则称为多叉树。如果样例的分类只有两种，那么该模型是二分类模型，否则是多分类模型。图 5-4 的决策树就是一个二叉树，且该模型是二分类模型。

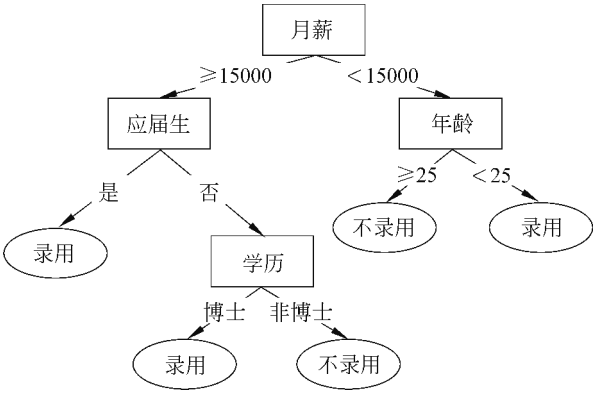
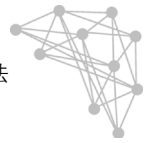


图 5-4　小张画出的决策树

小张以自身信息作为决策树的输入，从根节点“月薪”这一属性开始判断。小张要求月薪小于 15000，走向右分支到达内部节点年龄，继续判断；年龄小于 25 岁，到达叶节点录用，



得到分类结果。由此可知小张大概率会被录用。需要注意的是，一个属性可以在树的多个不同分支出现，如果该公司面对一个月薪要求大于或等于 15000 的博士应届生时，还要考虑年龄是否超过 40 岁，则决策过程可以表示为图 5-5 所示的模型。

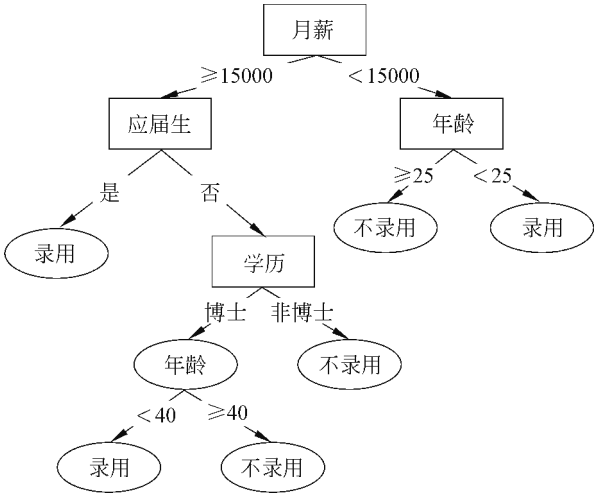


图 5-5　增加“年龄是否超过 40 岁”分支的决策树实例

有多种构造决策树的方法，这些方法都是从根节点出发，依次对属性进行分类。分类决策树的构造分为以下 3 步。

(1) 确定分割规则。确定划分的属性及其阈值，将数据划分成不相交的子集，再为每个子集确定划分属性。如图 5-5 所示，优先选择“月薪”这一属性，并以 15000 为阈值，将数据划分为大于或等于 15000 的 2、3、5、6 为一组，小于 15000 的 1、4 为一组；再为小于 15000 的一组选择“年龄”这一属性，并以 25 为阈值继续划分。

(2) 确定叶节点。确定当前节点是继续分割还是作为叶节点，判断的标准是：如果当前节点中每个成员都属于相同的类，就可以当作叶节点；否则继续选择属性，对该节点的成员进行划分。

(3) 把类别赋予叶节点。

然而是否一定要先从“月薪”开始分类？或者是否改变各分类属性的顺序后依然满足表 5-1 的数据？图 5-6 是由表 5-1 的数据归纳的另一种决策树。

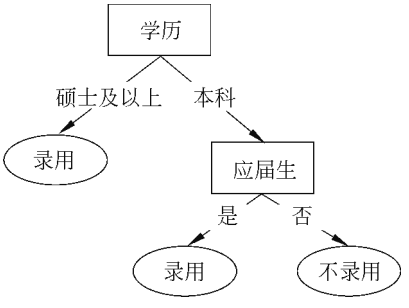


图 5-6　以“学历”属性为根节点的决策树实例

对于表 5-1 的数据，图 5-4 和图 5-6 的决策树都能够正确分类，图 5-6 的决策树仅用了两种属性进行判断，显然要高效得多。这说明在构建决策树时，优先选择合适的属性是十分重要的。那么如何选择继续分割节点成员的属性呢？

选择哪一个属性来划分当前节点成员直接决定决策树的结构。这就需要一种指标来评价每个属性，从中选取最优者。评价指标包括信息增益、信息增益率、基尼指数等，相应的决

策树算法主要有 ID3、C4.5、CART 等。

5.3.2 信息增益

熵是系统混乱程度的度量。熵可以表示任何一种能量在空间均匀分布的程度，分布越均匀，熵越大。比如对于同一堆树叶，随机散落的状态的熵比成堆状态的熵更大。

香农提出了信息的概念：信息是对不确定性的消除，比如有人指着一名小男孩说他不到 5 岁，这条信息消除了一部分对这个小男孩年龄的不确定性，使猜测缩小了范围。信息量是信息和先验知识的差距，比如有人告诉你一条已知的信息，那么这条信息是无用的，信息量为零，再比如“中国足球队打败了阿根廷足球队”无疑具有很大的信息量，因为它所描述的事件是小概率事件；相反，“阿根廷足球队打败了中国足球队”的信息量很小，因为它所描述的事件概率较大。因此，描述先验概率较小的事件发生的信息所含的信息量更大。香农基于先验概率来定义如下信息量公式：

$$I(x)=\log_a\left(\frac{1}{p}\right)=-\log_a(p) \tag{5-50}$$

其中，x 是消息描述的事件，p 是 x 的先验概率，a 一般取 2。

信息量描述的是信息源发出的单面事件消除的不确定性，没有考虑另一面的事件，不能描述信息源消除的平均不确定性，因此可以采用信息量的期望来描述，即信息熵：

$$H(X)=E[I(x_i)]=-\sum_{i=1}^{n} p_i \log_a p_i \tag{5-51}$$

对于决策树上某个节点的样本，这个度量反映了该节点上的特征对样本分类的不纯度。在实际应用时，可以用各类样本的比例作为概率的估计来计算样本的不纯度。例如，5 个样本分别属于不同类别，则信息熵为

$$H(x)=-(5\times 0.2\times \log_2 0.2)\approx 2.3219$$

此时，样本不纯度最大，不确定性最大。如果 5 个样本同属一类，则信息熵为

$$H(x)=-(1\times \log_2 1)=0$$

此时，样本最纯，没有不确定性。

例 5.3 计算表 5-1 中数据集的信息熵。

解：表 5-1 中的样本分为两类：录用和不录用各包含 3 个样本，根据式(5-5)，其信息熵为

$$H(X)=-\frac{3}{6}\log_2\frac{3}{6}-\frac{3}{6}\log_2\frac{3}{6}=1$$

最理想的属性能够直接将样本集合的各类区分，分开后的各子集都是同类，各子集的熵不纯度为零。当然，如果不存在这样的属性，那应当选择能够使不纯度最有效减少的属性。当把样本集 A 按照第 j 个属性划分为 n 个独立的子集 $A_1,A_2,\cdots,A_n$ 时，则 A 的信息熵为 n 个子集的信息熵按样本数量的比例作加权和：

$$H(A,F^{(J)})=\frac{|A_1|}{|A|}H(A_1)+\frac{|A_2|}{|A|}H(A_2)+\cdots+\frac{|A_n|}{|A|}H(A_n) \tag{5-52}$$

其中，$|A|$ 表示样本总个数，$|A_1|,|A_2|,\cdots,|A_n|$ 表示各子集样本个数。根据式(5-52)，可以计算一个样本集合选择了某个属性进行分类后的信息熵。

采用第 n 个属性划分样本集 A，划分后的熵不纯度比划分前的熵不纯度减少量为

$$\mathrm{Gain}(A,F^{(J)})=H(A)-H(A,F^{(J)}) \tag{5-53}$$

这个熵不纯度的减少量称为信息增益。

如果这个属性是最优的，那么分类后的信息熵相比分类前减少得最多，信息增益最大，各样本差异最大，最有利于分类。

基于上述分析，可以比较各属性分类后的信息增益来选择最优分类属性，即信息增益最大的是最优属性。将这一属性选择方法应用到决策树的构造中就是 ID3 决策树算法，其一般流程为：首先计算当前节点所有样本的信息熵，比较采用不同属性划分样本得到的信息增益，选择具有最大信息增益的属性赋予当前节点；如果子节点只包含一类样本，则该分支不再生长，该节点为叶节点；如果子节点仍包含不同类样本，则重复以上步骤，直到所有子节点都是叶节点为止。

例 5.4　分别计算表 5-1 中的数据选择“性别”和“学历”作为根节点划分属性的解：

选择“性别”作为属性时，

$$\begin{aligned}&\mathrm{Gain}(A,\text{性别})\\&=1+\left(\frac{3}{6}\times\left(\frac{1}{3}\times\log_2\frac{1}{3}+\frac{2}{3}\times\log_2\frac{2}{3}\right)+\frac{3}{6}\times\left(\frac{1}{3}\times\log_2\frac{1}{3}+\frac{2}{3}\times\log_2\frac{2}{3}\right)\right)\\&\approx 0.0817\end{aligned}$$

选择“学历”作为属性时，

$$\begin{aligned}&\mathrm{Gain}(A,\text{学历})\\&=1+\left(\frac{4}{6}\times\left(\frac{1}{4}\times\log_2\frac{1}{4}+\frac{3}{4}\times\log_2\frac{3}{4}\right)\right)\\&\approx 0.459\end{aligned}$$

5.3.3　信息增益率

采用信息增益选择属性存在缺点，算法会偏向取值情况多的属性，这一情况不利于分类。C4.5 决策树算法对此进行了改进，采用信息增益率作为选择属性的依据。信息增益率定义如下：

$$\mathrm{GainRatio}(A,F^{(j)})=\frac{\mathrm{Gain}(A,F^{(j)})}{\mathrm{SplitInfo}(F^{(j)})} \tag{5-54}$$

其中，$\mathrm{SplitInfo}(F^{(j)})$ 称为划分信息，定义如下：

$$\mathrm{SplitInfo}(F^{(j)})=-\sum_{i=1}^{N}\frac{|A_i|}{|A|}\log_2\frac{|A_i|}{|A|} \tag{5-55}$$

其中，N 为采用属性 $F^{(j)}$ 进行划分时得到的子集数，$|A_i|$ 是第 i 个子集的样本数。

在一些情况下，样本子集数增加，$\mathrm{SplitInfo}(F^{(j)})$ 也会增加，这在一定程度上抑制了样本子集数多时信息增益过大的趋势。但采用信息增益率作为属性选择依据也有缺点：算法会偏向取值情况少的属性。所以 C4.5 决策树算法的策略是先选出信息增益高于平均值的一批属性，再从中选择信息增益率最高的属性。

例 5.5　计算将“应届生”作为根节点划分属性的信息增益率。

解：信息增益率计算如下：

$$\text{SplitInfo}(\text{应届生}) = -\frac{2}{6}\log_2\frac{2}{6} - \frac{4}{6}\log_2\frac{4}{6} \approx 0.918$$

$$\text{GainRatio}(A,\text{应届生}) = \frac{\text{Gain}(A,\text{应届生})}{\text{SplitInfo}(\text{应届生})} = 0.5$$

5.3.4 基尼指数

CART 决策树采用基尼指数来选择划分属性。对于有 k 个类别的样本集 A，假设样本属于第 k 类的概率为 P_k，则此样本集的纯度可用基尼指数来度量：

$$\text{Gini}(A) = 1 - \sum_{k=1}^{K} {p_k}^2 \tag{5-56}$$

在表 5-1 的数据中，3 个被录用，3 个不被录用，该样本集 B 的基尼指数为

$$\text{Gini}(B) = 1 - \left[\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2\right] = 0.5$$

假如有 1 个被录用，5 个不被录用，则该样本集 B 的基尼指数为

$$\text{Gini}(B) = 1 - \left[\left(\frac{1}{6}\right)^2 + \left(\frac{5}{6}\right)^2\right] \approx 0.278$$

假如全部被录用，则该样本集 B 的基尼指数为

$$\text{Gini}(B) = 1 - 1^2 = 0$$

通过比较以上情况的基尼指数，可以发现样本集的纯度越高，基尼指数越小。当集合内的样本属于同一类别时，基尼指数达到最小值零。所以，基尼指数是一种样本纯度的度量指标。与信息熵类似，采用第 j 个属性划分样本集 A，该属性的基尼指数为各子集的基尼指数按样本数量的比例作加权和：

$$\text{Gini}_{\text{index}}(A, F^{(j)}) = \sum_{i=1}^{N} \frac{|A_i|}{|A|}\text{Gini}(A_i) \tag{5-57}$$

选择划分属性时，选择使划分前后基尼指数减少最多的属性，或者说使划分后基尼指数最小的属性。

例 5.6 计算表 5-1 中将“学历”作为根节点划分属性的基尼指数。

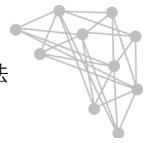
解：基尼指数计算如下：

$$\text{Gini}(A,\text{学历}) = \frac{4}{6}\left(1 - \left[\left(\frac{1}{4}\right)^2 + \left(\frac{3}{4}\right)^2\right]\right) \approx 0.25$$

5.3.5 剪枝处理

决策树是充分考虑了所有训练样本而生成的复杂树，它在学习的过程中为了尽可能地正确分类训练样本，需要不停地对节点进行划分，这会导致整棵树的分支过多，造成决策树很庞大。决策树庞大有可能导致在训练集上表现很好，但在测试数据上的表现与训练数据差别很大，即过拟合的情况。决策树越复杂，过拟合的程度越高。所以，为了避免过拟合，需要对决策树进行剪枝。一般情况下，有两种剪枝策略，分别是预剪枝和后剪枝。

预剪枝就是控制决策树的生长。在决策树的生长过程中，每个节点在划分前，先对其进行估计，如果该节点的划分不能提升决策树的泛化性能，那么不再划分该节点，并设为叶节



点。对于决策树的泛化性能，可以将数据集分为训练集和测试集，使用节点划分前后决策树在测试集上的正确率来体现泛化性能。表 5-2 和表 5-3 的苹果数据集详细说明了剪枝过程。

表 5-2　苹果训练集

编　　号	颜　　色	表　　皮	硬　　度	大　　小	好　　果
1	深红	磕碰	软	大	是
2	浅绿	完整	硬	大	否
3	深红	磕碰	软	小	是
4	浅绿	磕碰	硬	小	否
5	浅红	完整	软	小	是
6	深红	完整	硬	小	否
7	浅绿	磕碰	硬	大	否
8	浅红	完整	软	小	是
9	浅绿	磕碰	软	小	否
10	深红	完整	软	大	是

表 5-3　苹果验证集

编　　号	颜　　色	表　　皮	硬　　度	大　　小	好　　果
11	深红	完整	软	大	是
12	浅绿	完整	软	大	否
13	浅红	完整	软	大	否
14	深红	磕碰	软	小	是
15	浅绿	完整	硬	小	否
16	浅绿	磕碰	软	小	否
17	深红	完整	软	大	是

首先，采用信息增益选择根节点的划分属性。根据式(5-53)计算按各属性划分后的信息增益。划分前的信息熵为

$$H(A)=-\frac{5}{10}\log_2\frac{5}{10}-\frac{5}{10}\log_2\frac{5}{10}=1$$

则采用“颜色”这一属性划分后的信息增益为

$$\text{Gain}(A,\text{颜色})=1-\frac{4}{10}\times\left(-\frac{3}{4}\log_2\frac{3}{4}-\frac{1}{4}\log_2\frac{1}{4}\right)\approx 0.6755$$

采用“表皮”这一属性划分后的信息增益为

$$\text{Gain}(A,\text{表皮})=1-\frac{5}{10}\times\left(-\frac{2}{5}\log_2\frac{2}{5}-\frac{3}{5}\log_2\frac{3}{5}\right)\times 2\approx 0.0290$$

采用“硬度”这一属性划分后的信息增益为

$$\text{Gain}(A,\text{硬度})=1-\frac{6}{10}\times\left(-\frac{5}{6}\log_2\frac{5}{6}-\frac{1}{6}\log_2\frac{1}{6}\right)\approx 0.6100$$

采用“大小”这一属性划分后的信息增益为

$$\text{Gain}(A,\text{大小})=1-1=0$$

可以看出，采用“颜色”划分的信息增益最大，所以考虑根节点的属性为“颜色”。接下来用测试集验证该节点的划分能否提升决策树的泛化性能。

在划分前，所有样本集中在根节点。节点会被标记为训练集中样本数最多的类别，当各类比例相等时，可选任意一类，这里根节点的类别选择为“坏果”。用验证集对当前单节点决策树进行评估，有 4 个样本被分类正确，所以验证集精度为 57.1%。用“颜色”这一属性划分后得到 3 个节点，分别对应“深红”、“浅红”和“浅绿”。根据训练集的样本类别，节点类别分别为“好果”、“好果”和“坏果”。在验证集中，除编号 13 的样本外全部分类正确，验证集精度为 85.71%，大于 57.1%，由此证明决策树的泛化性能提升了，此处无须剪枝。继续对新生的 3 个节点选择划分属性，并判断决策树的泛化性能是否提升，直到所有节点为叶节点。

预剪枝使得决策树部分分支没有生长，这在一定程度上降低了过拟合的风险。但有些分支的生长虽然不能提升决策树的泛化性能，其后续的分支有可能提升泛化性能。预剪枝使这种类型的分支无法生长，导致决策树有欠拟合的风险。

后剪枝是在决策树生长完后再对其进行修剪。其核心思想是：从叶节点出发，如果减去具有相同父节点的叶节点，会使决策树的泛化性能提升，则执行剪枝，并将父节点作为新的叶节点。以回溯的形式剪去不必要的节点，直到没有可剪枝的节点为止。表 5-2 的苹果数据集不进行剪枝，则可生成图 5-7 所示的决策树。

接下来对图 5-7 所示决策树进行后剪枝操作，首先对最下层的两个叶节点进行判断。剪枝前，决策树仅对编号 13 的样本分类错误，在验证集上的精度为 85.71%。若剪去最下层的两个叶节点，根据训练集中“深红”类别的样本占多数的情况，所以该节点类别为“是”。图 5-7 决策树经过一次后剪枝操作得到的决策树如图 5-8 所示。

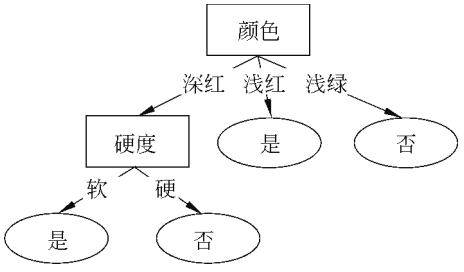


图 5-7　苹果数据集不剪枝时对应的决策树

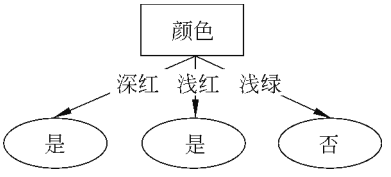


图 5-8　进行一次后剪枝操作的决策树

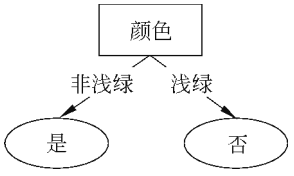


图 5-9　合并相同类别叶节点的决策树

剪枝后的决策树在验证集上的精度为 85.71%，与没有剪枝的决策树的精度相同，决策树的泛化性能没有变化。这种情况下，选择更简洁的决策树，同时合并同一父节点下相同类别的叶节点，得到图 5-9 所示的决策树。

对于当前决策树，继续判断是否需要剪枝。如果继续剪枝，决策树仅剩根节点，根据训练集样本类别数，根节点类别为“是”或“否”都可以，在验证集上的精度分别为 42.85%和57.14%，均

小于剪枝前的 85.71%，所以当前决策树不需要剪枝。

一般情况下，后剪枝决策树比预剪枝决策树拥有更多分支，欠拟合风险更小，泛化性能更好。但由于后剪枝前需要先生成完整决策树，所以后剪枝决策树的训练时间更久。

5.3.6　连续值处理

现实中常遇到属性的取值为连续值的情况。连续值属性的可取值数是无限的，无法直接根据连续值属性的可取值来确定划分界限，需要对连续值属性做离散化操作。C4.5 采用二分法对连续值属性进行处理。

对于一个样本集 A，其样本在连续属性 c 上有从小到大 n 个不同取值 $c_1,c_2,\cdots,c_n$。选取相邻两个取值的均值作为划分点 t_i（如式(5-58)），有 $n-1$ 种选择。

$$t_i=\frac{c_i+c_{i+1}}{2},\quad 1\leqslant i\leqslant n-1 \tag{5-58}$$

任意一种选择都可以作为候选划分点，将 c 的取值划分为两部分。然后选取一种评估指标得到最优划分点。比如各候选划分点划分后的信息增益。

$$\mathrm{Gain}(A,c)=\max_{1\leqslant i\leqslant n-1}\mathrm{Gain}(A,c,t_i) \tag{5-59}$$

其中，$\mathrm{Gain}(A,c,t_i)$ 是样本集 A 选择划分点 t_i 划分后的信息增益，最终选择使 $\mathrm{Gain}(A,c,t_i)$ 最大的划分点。

5.3.7　缺失值处理

现实中常遇到样本的某些属性缺失的问题。抛弃少量的不完整的样本对决策树的创建影响不大，但如果属性缺失较多或关键属性缺失，得到的决策树是不完整的，而且可能带来错误信息。如果直接抛弃不完整的样本，只使用完整的样本进行学习，将浪费数据集中大量的信息。因此，有必要利用存在缺失值的样本进行训练。缺失值问题可以从以下 3 方面考虑。

(1) 样本存在缺失值，如何选择划分属性？

假定训练集 A 存在属性 c，属性 c 有 N 个取值 $c_1,c_2,\cdots,c_n$。$\widetilde{A}$ 表示在属性 c 上没有缺失值的样本子集。可以根据 $\widetilde{A}$ 来判断是否采用属性 c 进行划分。$\widetilde{A^n}$ 表示 $\widetilde{A}$ 在属性 c 取值为 n 的子集，$\widetilde{A_k}$ 表示 $\widetilde{A}$ 中属于第 k 类（$k=1,2,\cdots,y$）的样本子集，则 $\widetilde{A}$ 为每个样本赋予一个权重 w_a，并定义

$$p=\frac{\sum\limits_{a\in\widetilde{A}}w_a}{\sum\limits_{a\in A}w_a} \tag{5-60}$$

$$\widetilde{p_k}=\frac{\sum\limits_{a\in\widetilde{A_k}}w_a}{\sum\limits_{a\in\widetilde{A}}w_a}(1\leqslant k\leqslant y) \tag{5-61}$$

$$\widetilde{r_n}=\frac{\sum\limits_{a\in\widetilde{A^n}} w_a}{\sum\limits_{a\in\widetilde{A}} w_a}(1\leqslant n\leqslant N) \tag{5-62}$$

其中，p 表示完整样本占整个数据集的比例，$\widetilde{p_k}$ 表示完整样本中第 k 类的比例，$\widetilde{r_n}$ 表示完整样本中属性 c 取值为 c_n 的比例。由此，信息增益公式可演变为

$$\mathrm{Gain}(A,c)=p\times\mathrm{Gain}(\tilde{A},c)=p\times(H(\tilde{A})-\sum_{n=1}^{N}\tilde{r}_n H(\widetilde{A^n})) \tag{5-63}$$

$$H(\tilde{A})=-\sum_{k=1}^{y}\widetilde{p_k}\log_2\widetilde{p_k} \tag{5-64}$$

从公式上看，计算存在缺失值样本的划分信息增益时，先忽略属性值缺失的样本，然后计算的值乘上完整样本占整个数据集的比例。

(2) 选择划分属性后，如何对训练集中属性不完整的样本分类？

如果样本 A 在划分属性 c 上的取值缺失，则将该样本分配到所有子节点中，权重 w_a 需乘上在属性 c 上有值的样本占划分成的子集样本个数的比例 $\widetilde{r_n}$。同时，计算错误率的时候，需要考虑样本权重。比如该节点是根据 c 属性划分，但是待分类样本 c 属性缺失，假设 c 属性离散，有两种取值，那么就把该待分类样本分配到两个子节点中去，但是权重需要乘以属性离散值对应的样本占划分成的子集样本的比例。

(3) 训练结束后，如何对属性值不完整的验证集样本分类？

分类时，如果待分类样本有缺失属性，而决策树决策过程中没有用到这些属性，则决策过程和没有缺失的样本一样；如果决策需要用到缺失属性，决策树可以在当前节点做多数投票来决定，即选择样本数最多的属性值。或者计算每个类别的概率，则待分类样本判定为最大概率对应的类别。

5.4 随机森林

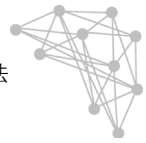
处理高维数据和大数据集时，会生成一个巨大的决策树，大量的内部节点意味着要反复计算属性选择指标，显然是不合适的。同时，得到的结构是基于特定样本集的，这个样本集只是所有可能情况的一次随机取样。受这种随机性的影响，得到的决策树具有一定偶然性。

针对此类问题，统计学家提出一种叫作 BootStrap 的策略，其基本思想是通过对现有样本进行有放回的采样来产生多个样本集，模拟数据的随机性，并在最后的结果中考虑随机性的影响。这种策略在模式识别问题中有大量应用，其中具有代表性的一种算法叫作随机森林算法。

5.4.1 基本概念

随机森林方法是从样本集中构建多个较小的决策树，通过多个树投票进行预测。以下为随机森林算法的一般步骤。

(1) 从样本集中有放回地随机选取 m 个样本组成样本集。



（2）用随机选取的样本集构造一个决策树。构造时，从所有属性中随机选择 k 个属性，不放回，k 通常取 $\log_2 K$，K 为所有属性的总数。

（3）依靠抽取的样本子集和属性生成决策树。

（4）重复以上过程 n 次，生成 n 个决策树。

（5）用每个决策树对样本分类，通过多棵树投票进行预测。

图 5-10 所示为随机森林的原理图。

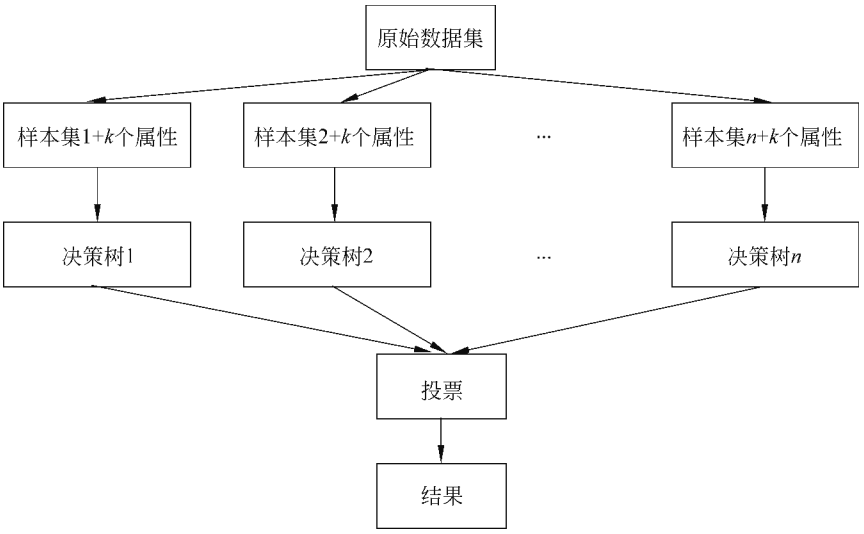


图 5-10　随机森林的原理图

随机森林算法既对训练样本进行了采样，又对属性进行了采样，充分保证了每棵决策树间的独立性。事实证明，这种方法具有能够有效提升模型的泛化能力，且保持较高的准确率；能够处理高维数据集；能够评估各属性的重要性等优势，在实践中得到了广泛应用。

5.4.2　袋外错误率

随机森林分类效果（错误率）与两个因素有关：一是森林中任意两棵树的相关性，相关性越大，错误率越大；二是森林中每棵树的分类能力，每棵树的分类能力越强，整个森林的错误率越低。

减小属性选择个数 k，树的相关性和分类能力也会相应的降低；增大 k，两者也会随之增大。所以关键问题是如何选择最优的 k（或者是范围）。要解决这个问题，主要依据计算袋外错误率（Out-Of-Bag Error）。

生成决策树时，从整个样本集中有放回地随机选取 m 个样本组成训练样本集，这种采样方式会导致约有 36％的样本永远不会被采样到，这些样本称为袋外样本。对于已经生成的随机森林，可以使用袋外样本测试其性能，将袋外样本输入已经生成的随机森林分类器，分类器会给出相应的分类结果，统计随机森林分类器分类错误的数目，分类错误的样本数占袋外样本数的比例就是袋外错误率。袋外错误率是无偏的，所以在随机森林算法中不需要再进行交叉验证或者使用单独的测试集来获取测试集误差的无偏估计。

5.5 Boosting 方法

Boosting 是一种将弱学习器提升为强学习器的算法，通过训练基学习器并根据其表现，对训练样本分布进行调整来实现。这个过程重复进行，直至基学习器数目达到事先指定的值 T，最终将这 T 个基学习器进行加权结合。Boosting 方法采用迭代过程对分类器的输入和输出进行加权处理，而不是简单地对其输出进行投票决策，通过融合多个分类器进行决策，可以大大提高分类器的性能。在每一次迭代过程中，根据分类的情况对各个样本进行加权。Boosting 方法在分类问题中的应用广泛且有效，它通过改变训练样本的权重学习多个分类器，并将这些分类器进行线性组合来提高分类器的性能。

AdaBoost 算法是 Boosing 方法中最具有代表性的算法。对于 Boosting 方法而言，需要解决两个问题：一是在每一轮训练中如何改变训练数据的权值或概率分布；二是如何将多个弱分类器组合成一个强分类器。对于第一个问题，AdaBoost 算法在每一轮训练中提高前一轮被弱分类器错误分类的样本的权值，使其受到更多的关注，同时降低那些被正确分类样本的权值。对于第二个问题，AdaBoost 采用“加性模型”，即对弱分类器通过加权多数表决的方法进行组合。

$$H(x)=\sum_{t=1}^{T}\alpha_t h_t(x) \tag{5-65}$$

其中，$H(x)$ 代表强分类器，α_t 代表第 t 个弱分类器的权值，$h_t(x)$ 代表第 t 个弱分类器。

对于 AdaBoost 算法，假设给定一个二分类的训练数据集为 D

$$D=\{(x_1,y_1),(x_2,y_2),\cdots,(x_t,y_t)\} \tag{5-66}$$

其中，每个样本点由实例和标签组成，x_i 代表实例，y_i 代表标签。

AdaBoost 算法步骤如下。

(1) 初始化训练数据的权值分布，N 为样本数量。

$$W_1=(w_{1,1},w_{1,2},\cdots,w_{1,i},\cdots,w_{1,N}),\quad w_{1i}=\frac{1}{N},\quad i=1,2,\cdots,N \tag{5-67}$$

(2) 对于 $t=1,2,\cdots,T$，其中 T 为训练轮数，执行如下循环。

① 使用具有初始权值的训练数据集学习，得到基本分类器：

$$h_t(x),x\to\{-1,+1\} \tag{5-68}$$

② 计算 $h_t(x)$ 在训练集上的分类误差：

$$e_t=\sum_{i=1}^{N}P(h_t(x_i)\neq y_i) \tag{5-69}$$

如果 $e_t>0.5$，说明此基学习器的性能低于 50%，直接进入下一个循环。

③ 计算 $h_t(x)$ 的系数：

$$\alpha_t=\frac{1}{2}\ln\left(\frac{1-e_t}{e_t}\right) \tag{5-70}$$

④ 更新训练数据集的权值分布：

$$w_{t+1,i}=\frac{w_{t,i}}{z_t}\exp(-\alpha_t y_i h_t(x_i)),\quad i=1,2,\cdots,N \tag{5-71}$$