第 4 章　朴素贝叶斯法

朴素贝叶斯（naïve Bayes）法是基于贝叶斯定理与特征条件独立假设的分类方法[①]. 对于给定的训练数据集，首先基于特征条件独立假设学习输入/输出的联合概率分布；然后基于此模型，对给定的输入 x，利用贝叶斯定理求出后验概率最大的输出 y. 朴素贝叶斯法实现简单，学习与预测的效率都很高，是一种常用的方法.

本章叙述朴素贝叶斯法，包括朴素贝叶斯法的学习与分类、朴素贝叶斯法的参数估计算法.

4.1　朴素贝叶斯法的学习与分类

4.1.1　基本方法

设输入空间 $\mathcal{X}\subseteq\mathbf{R}^n$ 为 n 维向量的集合，输出空间为类标记集合 $\mathcal{Y}=\{c_1, c_2,\cdots,c_K\}$. 输入为特征向量 $x\in\mathcal{X}$，输出为类标记（class label）$y\in\mathcal{Y}$. X 是定义在输入空间 $\mathcal{X}$ 上的随机向量，Y 是定义在输出空间 $\mathcal{Y}$ 上的随机变量. $P(X,Y)$ 是 X 和 Y 的联合概率分布. 训练数据集

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

由 $P(X,Y)$ 独立同分布产生.

朴素贝叶斯法通过训练数据集学习联合概率分布 $P(X,Y)$. 具体地，学习以下先验概率分布及条件概率分布. 先验概率分布

$$P(Y=c_k),\quad k=1,2,\cdots,K \tag{4.1}$$

条件概率分布

$$P(X=x\,|\,Y=c_k)=P(X^{(1)}=x^{(1)},\cdots,X^{(n)}=x^{(n)}\,|\,Y=c_k),\quad k=1,2,\cdots,K \tag{4.2}$$

于是学习到联合概率分布 $P(X,Y)$.

条件概率分布 $P(X=x\,|\,Y=c_k)$ 有指数级数量的参数，其估计实际是不可行的. 事实上，假设 $x^{(j)}$ 可取值有 S_j 个，$j=1,2,\cdots,n$，Y 可取值有 K 个，那么参数个数为 $K\prod_{j=1}^{n}S_j$.

① 注意：朴素贝叶斯法与贝叶斯估计（Bayesian estimation）是不同的概念.

朴素贝叶斯法对条件概率分布作了条件独立性的假设．由于这是一个较强的假设，朴素贝叶斯法也由此得名．具体地，条件独立性假设是

$$\begin{aligned} P(X=x \mid Y=c_k) &= P(X^{(1)}=x^{(1)},\cdots,X^{(n)}=x^{(n)} \mid Y=c_k) \\ &= \prod_{j=1}^{n} P(X^{(j)}=x^{(j)} \mid Y=c_k) \end{aligned} \tag{4.3}$$

朴素贝叶斯法实际上学习到生成数据的机制，所以属于生成模型．条件独立假设等于是说用于分类的特征在类确定的条件下都是条件独立的．这一假设使朴素贝叶斯法变得简单，但有时会牺牲一定的分类准确率．

朴素贝叶斯法分类时，对给定的输入 x，通过学习到的模型计算后验概率分布 $P(Y=c_k \mid X=x)$，将后验概率最大的类作为 x 的类输出．后验概率计算根据贝叶斯定理进行：

$$P(Y=c_k \mid X=x)=\frac{P(X=x \mid Y=c_k)P(Y=c_k)}{\sum_k P(X=x \mid Y=c_k)P(Y=c_k)} \tag{4.4}$$

将式 (4.3) 代入式 (4.4) 有

$$P(Y=c_k \mid X=x)=\frac{P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k)}{\sum_k P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k)}, \quad k=1,2,\cdots,K \tag{4.5}$$

这是朴素贝叶斯法分类的基本公式．于是，朴素贝叶斯分类器可表示为

$$y=f(x)=\arg\max_{c_k}\frac{P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k)}{\sum_k P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k)} \tag{4.6}$$

注意到，在式 (4.6) 中分母对所有 c_k 都是相同的，所以，

$$y=\arg\max_{c_k} P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)} \mid Y=c_k) \tag{4.7}$$

4.1.2 后验概率最大化的含义

朴素贝叶斯法将实例分到后验概率最大的类中．这等价于期望风险最小化．假设选择 0-1 损失函数：

$$L(Y,f(X))=\begin{cases}1, & Y \neq f(X)\\ 0, & Y=f(X)\end{cases}$$

式中 $f(X)$ 是分类决策函数．这时，期望风险函数为

$$R_{\exp}(f)=E[L(Y,f(X))]$$

期望是对联合分布 $P(X,Y)$ 取的. 由此取条件期望

$$R_{\exp}(f)=E_X\sum_{k=1}^{K}[L(c_k,f(X))]P(c_k\mid X)$$

为了使期望风险最小化，只需对 $X=x$ 逐个极小化，由此得到：

$$\begin{aligned}f(x)&=\arg\min_{y\in\mathcal{Y}}\sum_{k=1}^{K}L(c_k,y)P(c_k\mid X=x)\\&=\arg\min_{y\in\mathcal{Y}}\sum_{k=1}^{K}P(y\neq c_k\mid X=x)\\&=\arg\min_{y\in\mathcal{Y}}(1-P(y=c_k\mid X=x))\\&=\arg\max_{y\in\mathcal{Y}}P(y=c_k\mid X=x)\end{aligned}$$

这样一来，根据期望风险最小化准则就得到了后验概率最大化准则：

$$f(x)=\arg\max_{c_k}P(c_k\mid X=x)$$

即朴素贝叶斯法所采用的原理.

4.2 朴素贝叶斯法的参数估计

4.2.1 极大似然估计

在朴素贝叶斯法中，学习意味着估计 $P(Y=c_k)$ 和 $P(X^{(j)}=x^{(j)}\mid Y=c_k)$. 可以应用极大似然估计法估计相应的概率. 先验概率 $P(Y=c_k)$ 的极大似然估计是

$$P(Y=c_k)=\frac{\sum_{i=1}^{N}I(y_i=c_k)}{N},\quad k=1,2,\cdots,K \tag{4.8}$$

设第 j 个特征 $x^{(j)}$ 可能取值的集合为 $\{a_{j1},a_{j2},\cdots,a_{jS_j}\}$，条件概率 $P(X^{(j)}=a_{jl}\mid Y=c_k)$ 的极大似然估计是

$$P(X^{(j)}=a_{jl}\mid Y=c_k)=\frac{\sum_{i=1}^{N}I(x_i^{(j)}=a_{jl},y_i=c_k)}{\sum_{i=1}^{N}I(y_i=c_k)}$$

$$j=1,2,\cdots,n;\quad l=1,2,\cdots,S_j;\quad k=1,2,\cdots,K \tag{4.9}$$

式中，$x_i^{(j)}$ 是第 i 个样本的第 j 个特征；a_{jl} 是第 j 个特征可能取的第 l 个值；I 为指示函数.

4.2.2 学习与分类算法

下面给出朴素贝叶斯法的学习与分类算法.

算法 4.1（朴素贝叶斯算法（naïve Bayes algorithm））

输入：训练数据 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中 $x_i=(x_i^{(1)},x_i^{(2)},\cdots,x_i^{(n)})^{\mathrm{T}}$，$x_i^{(j)}$ 是第 i 个样本的第 j 个特征，$x_i^{(j)}\in\{a_{j1},a_{j2},\cdots,a_{jS_j}\}$，$a_{jl}$ 是第 j 个特征可能取的第 l 个值，$j=1,2,\cdots,n$，$l=1,2,\cdots,S_j$，$y_i\in\{c_1,c_2,\cdots,c_K\}$；实例 x；

输出：实例 x 的分类.

（1）计算先验概率及条件概率

$$P(Y=c_k)=\frac{\sum_{i=1}^{N}I(y_i=c_k)}{N},\quad k=1,2,\cdots,K$$

$$P(X^{(j)}=a_{jl}\mid Y=c_k)=\frac{\sum_{i=1}^{N}I(x_i^{(j)}=a_{jl},y_i=c_k)}{\sum_{i=1}^{N}I(y_i=c_k)}$$

$$j=1,2,\cdots,n;\quad l=1,2,\cdots,S_j;\quad k=1,2,\cdots,K$$

（2）对于给定的实例 $x=(x^{(1)},x^{(2)},\cdots,x^{(n)})^{\mathrm{T}}$，计算

$$P(Y=c_k)\prod_{j=1}^{n}P(X^{(j)}=x^{(j)}\mid Y=c_k),\quad k=1,2,\cdots,K$$

（3）确定实例 x 的类

$$y=\arg\max_{c_k}P(Y=c_k)\prod_{j=1}^{n}P(X^{(j)}=x^{(j)}\mid Y=c_k)$$

■

例 4.1 试由表 4.1 的训练数据学习一个朴素贝叶斯分类器并确定 $x=(2,S)^{\mathrm{T}}$ 的类标记 y. 表中 $X^{(1)}$，$X^{(2)}$ 为特征，取值的集合分别为 $A_1=\{1,2,3\}$，$A_2=\{S,M,L\}$，Y 为类标记，$Y\in C=\{1,-1\}$.

表 4.1 训练数据

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$X^{(1)}$	1	1	1	1	1	2	2	2	2	2	3	3	3	3	3
$X^{(2)}$	S	M	M	S	S	S	M	M	L	L	L	M	M	L	L
Y	−1	−1	1	1	−1	−1	−1	1	1	1	1	1	1	1	−1

解 根据算法 4.1，由表 4.1，容易计算下列概率：

$$P(Y=1)=\frac{9}{15},\quad P(Y=-1)=\frac{6}{15}$$

$$P(X^{(1)}=1\mid Y=1)=\frac{2}{9},\quad P(X^{(1)}=2\mid Y=1)=\frac{3}{9},\quad P(X^{(1)}=3\mid Y=1)=\frac{4}{9}$$

$$P(X^{(2)}=S\mid Y=1)=\frac{1}{9},\quad P(X^{(2)}=M\mid Y=1)=\frac{4}{9},\quad P(X^{(2)}=L\mid Y=1)=\frac{4}{9}$$

$$P(X^{(1)}=1\mid Y=-1)=\frac{3}{6},\quad P(X^{(1)}=2\mid Y=-1)=\frac{2}{6},\quad P(X^{(1)}=3\mid Y=-1)=\frac{1}{6}$$

$$P(X^{(2)}=S\mid Y=-1)=\frac{3}{6},\quad P(X^{(2)}=M\mid Y=-1)=\frac{2}{6},\quad P(X^{(2)}=L\mid Y=-1)=\frac{1}{6}$$

对于给定的 $x=(2,S)^{\mathrm{T}}$ 计算：

$$P(Y=1)P(X^{(1)}=2\mid Y=1)P(X^{(2)}=S\mid Y=1)=\frac{9}{15}\cdot\frac{3}{9}\cdot\frac{1}{9}=\frac{1}{45}$$

$$P(Y=-1)P(X^{(1)}=2\mid Y=-1)P(X^{(2)}=S\mid Y=-1)=\frac{6}{15}\cdot\frac{2}{6}\cdot\frac{3}{6}=\frac{1}{15}$$

因为 $P(Y=-1)P(X^{(1)}=2\mid Y=-1)P(X^{(2)}=S\mid Y=-1)$ 最大，所以 $y=-1$. ■

4.2.3 贝叶斯估计

用极大似然估计可能会出现所要估计的概率值为 0 的情况. 这时会影响到后验概率的计算结果，使分类产生偏差. 解决这一问题的方法是采用贝叶斯估计. 具体地，条件概率的贝叶斯估计是

$$P_\lambda(X^{(j)}=a_{jl}\mid Y=c_k)=\frac{\sum_{i=1}^{N}I(x_i^{(j)}=a_{jl},y_i=c_k)+\lambda}{\sum_{i=1}^{N}I(y_i=c_k)+S_j\lambda} \tag{4.10}$$

式中 $\lambda\geqslant 0$. 等价于在随机变量各个取值的频数上赋予一个正数 $\lambda>0$. 当 $\lambda=0$ 时就是极大似然估计. 常取 $\lambda=1$，这时称为拉普拉斯平滑（Laplace smoothing）. 显然，对任何 $l=1,2,\cdots,S_j$，$k=1,2,\cdots,K$，有

$$P_\lambda(X^{(j)}=a_{jl}\mid Y=c_k)>0$$

$$\sum_{l=1}^{S_j}P(X^{(j)}=a_{jl}\mid Y=c_k)=1$$

表明式 (4.10) 确为一种概率分布. 同样，先验概率的贝叶斯估计是

$$P_\lambda(Y=c_k)=\frac{\sum_{i=1}^{N}I(y_i=c_k)+\lambda}{N+K\lambda} \tag{4.11}$$

例 4.2 问题同例 4.1，按照拉普拉斯平滑估计概率，即取 $\lambda=1$.

解 $A_1=\{1,2,3\}$，$A_2=\{S,M,L\}$，$C=\{1,-1\}$. 按照式 (4.10) 和式 (4.11) 计算下列概率：

$$P(Y=1)=\frac{10}{17},\quad P(Y=-1)=\frac{7}{17}$$

$$P(X^{(1)}=1\mid Y=1)=\frac{3}{12},\quad P(X^{(1)}=2\mid Y=1)=\frac{4}{12},\quad P(X^{(1)}=3\mid Y=1)=\frac{5}{12}$$

$$P(X^{(2)}=S\mid Y=1)=\frac{2}{12},\quad P(X^{(2)}=M\mid Y=1)=\frac{5}{12},\quad P(X^{(2)}=L\mid Y=1)=\frac{5}{12}$$

$$P(X^{(1)}=1\mid Y=-1)=\frac{4}{9},\quad P(X^{(1)}=2\mid Y=-1)=\frac{3}{9},\quad P(X^{(1)}=3\mid Y=-1)=\frac{2}{9}$$

$$P(X^{(2)}=S\mid Y=-1)=\frac{4}{9},\quad P(X^{(2)}=M\mid Y=-1)=\frac{3}{9},\quad P(X^{(2)}=L\mid Y=-1)=\frac{2}{9}$$

对于给定的 $x=(2,S)^{\mathrm{T}}$ 计算：

$$P(Y=1)P(X^{(1)}=2\mid Y=1)P(X^{(2)}=S\mid Y=1)=\frac{10}{17}\cdot\frac{4}{12}\cdot\frac{2}{12}=\frac{5}{153}=0.0327$$

$$P(Y=-1)P(X^{(1)}=2\mid Y=-1)P(X^{(2)}=S\mid Y=-1)=\frac{7}{17}\cdot\frac{3}{9}\cdot\frac{4}{9}=\frac{28}{459}=0.0610$$

由于 $P(Y=-1)P(X^{(1)}=2\mid Y=-1)P(X^{(2)}=S\mid Y=-1)$ 最大，所以 $y=-1$. ■

本 章 概 要

1. 朴素贝叶斯法是典型的生成学习方法. 生成方法由训练数据学习联合概率分布 $P(X,Y)$，然后求得后验概率分布 $P(Y\mid X)$. 具体来说，利用训练数据学习 $P(X\mid Y)$ 和 $P(Y)$ 的估计，得到联合概率分布：

$$P(X,Y)=P(Y)P(X\mid Y)$$

概率估计方法可以是极大似然估计或贝叶斯估计：

2. 朴素贝叶斯法的基本假设是条件独立性，

$$\begin{aligned}P(X=x\mid Y=c_k)&=P(X^{(1)}=x^{(1)},\cdots,X^{(n)}=x^{(n)}\mid Y=c_k)\\&=\prod_{j=1}^{n}P(X^{(j)}=x^{(j)}\mid Y=c_k)\end{aligned}$$

这是一个较强的假设. 由于这一假设，模型包含的条件概率的数量大为减少，朴素贝叶斯法的学习与预测大为简化. 因而朴素贝叶斯法高效，且易于实现. 其缺

点是分类的性能不一定很高.

3. 朴素贝叶斯法利用贝叶斯定理与学到的联合概率模型进行分类预测.

$$P(Y \mid X)=\frac{P(X,Y)}{P(X)}=\frac{P(Y)P(X \mid Y)}{\sum_{Y} P(Y)P(X \mid Y)}$$

将输入 x 分到后验概率最大的类 y.

$$y=\arg\max_{c_k} P(Y=c_k)\prod_{j=1}^{n} P(X_j=x^{(j)} \mid Y=c_k)$$

后验概率最大等价于 0-1 损失函数时的期望风险最小化.

继续阅读

朴素贝叶斯法的介绍可见文献[1, 2]. 朴素贝叶斯法中假设输入变量都是条件独立的，如果假设它们之间存在概率依存关系，模型就变成了贝叶斯网络，参见文献[3].

习 题

4.1 用极大似然估计法推出朴素贝叶斯法中的概率估计公式 (4.8) 及公式 (4.9).

4.2 用贝叶斯估计法推出朴素贝叶斯法中的概率估计公式 (4.10) 及公式 (4.11).

参考文献

[1] Mitchell TM. Chapter 1: Generative and discriminative classifiers: Naïve Bayes and logistic regression. In: Machine Learning. Draft, 2005. http://www.cs.cmu.edu/~tom/mlbook/NBayeslogReg.pdf

[2] Hastie T, Tibshirani R, Friedman J. The Elements of Statistical Learning. Data Mining, Inference, and Prediction. Springer-Verlag, 2001(中译本：统计学习基础——数据挖掘、推理与预测. 范明，柴玉梅，昝红英等译. 北京：电子工业出版社，2004)

[3] Bishop C. Pattern Recognition and Machine Learning, Springer, 2006

第5章 决 策 树

决策树（decision tree）是一种基本的分类与回归方法．本章主要讨论用于分类的决策树．决策树模型呈树形结构，在分类问题中，表示基于特征对实例进行分类的过程．它可以认为是 if-then 规则的集合，也可以认为是定义在特征空间与类空间上的条件概率分布．其主要优点是模型具有可读性，分类速度快．学习时，利用训练数据，根据损失函数最小化的原则建立决策树模型．预测时，对新的数据，利用决策树模型进行分类．决策树学习通常包括 3 个步骤：特征选择、决策树的生成和决策树的修剪．这些决策树学习的思想主要来源于由 Quinlan 在 1986 年提出的 ID3 算法和 1993 年提出的 C4.5 算法，以及由 Breiman 等人在 1984 年提出的 CART 算法．

本章首先介绍决策树的基本概念，然后通过 ID3 和 C4.5 介绍特征的选择、决策树的生成以及决策树的修剪，最后介绍 CART 算法．

5.1 决策树模型与学习

5.1.1 决策树模型

定义 5.1（决策树） 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（node）和有向边（directed edge）组成．结点有两种类型：内部结点（internal node）和叶结点（leaf node）．内部结点表示一个特征或属性，叶结点表示一个类．

用决策树分类，从根结点开始，对实例的某一特征进行测试，根据测试结果，将实例分配到其子结点；这时，每一个子结点对应着该特征的一个取值．如此递归地对实例进行测试并分配，直至达到叶结点．最后将实例分到叶结点的类中．

图 5.1 是一个决策树的示意图．图中圆和方框分别表示内部结点和叶结点．

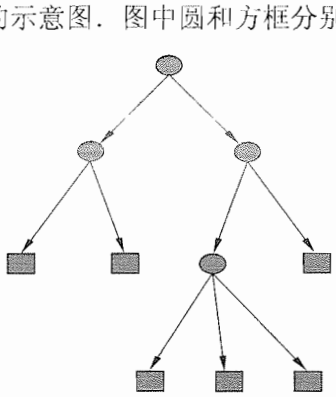

图 5.1 决策树模型

5.1.2 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．将决策树转换成 if-then 规则的过程是这样的：由决策树的根结点到叶结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 规则集合具有一个重要的性质：互斥并且完备．这就是说，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．这里所谓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．

5.1.3 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．这一条件概率分布定义在特征空间的一个划分（partition）上．将特征空间划分为互不相交的单元（cell）或区域（region），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．决策树的一条路径对应于划分中的一个单元．决策树所表示的条件概率分布由各个单元给定条件下类的条件概率分布组成．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 $P(Y|X)$．X 取值于给定划分下单元的集合，Y 取值于类的集合．各叶结点（单元）上的条件概率往往偏向某一个类，即属于某一类的概率较大．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．

图 5.2（a）示意地表示了特征空间的一个划分．图中的大正方形表示特征空间．这个大正方形被若干个小矩形分割，每个小矩形表示一个单元．特征空间划分上的单元构成了一个集合，X 取值为单元的集合．为简单起见，假设只有两类：正类和负类，即 Y 取值为+1 和−1．小矩形中的数字表示单元的类．图 5.2（b）示意地表示特征空间划分确定时，特征（单元）给定条件下类的条件概率分布．图 5.2（b）中条件概率分布对应于图 5.2（a）的划分．当某个单元 c 的条件概率满足 $P(Y=+1|X=c)>0.5$ 时，则认为这个单元属于正类，即落在这个单元的实例都被视为正例．图 5.2（c）为对应于图 5.2（b）中条件概率分布的决策树．

5.1.4 决策树学习

决策树学习，假设给定训练数据集

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

其中，$x_i=(x_i^{(1)},x_i^{(2)},\cdots,x_i^{(n)})^{\mathrm{T}}$ 为输入实例（特征向量），n 为特征个数，$y_i\in\{1,2,\cdots,K\}$ 为类标记，$i=1,2,\cdots,N$，N 为样本容量．学习的目标是根据给定的训练数据集构建一个决策树模型，使它能够对实例进行正确的分类．

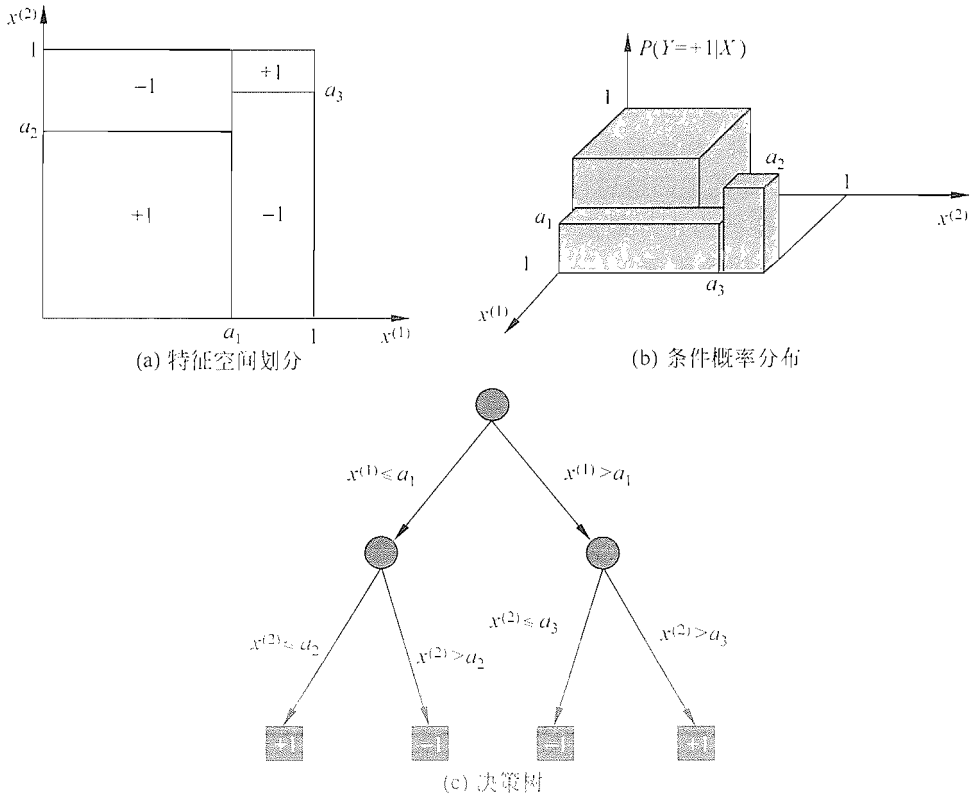

图 5.2 决策树对应于条件概率分布

决策树学习本质上是从训练数据集中归纳出一组分类规则. 与训练数据集不相矛盾的决策树（即能对训练数据进行正确分类的决策树）可能有多个，也可能一个也没有. 我们需要的是一个与训练数据矛盾较小的决策树，同时具有很好的泛化能力. 从另一个角度看，决策树学习是由训练数据集估计条件概率模型. 基于特征空间划分的类的条件概率模型有无穷多个. 我们选择的条件概率模型应该不仅对训练数据有很好的拟合，而且对未知数据有很好的预测.

决策树学习用损失函数表示这一目标. 如下所述，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. 因为从所有可能的决策树中选取最优决策树是 NP 完全问题，所以现实中决策树学习算法通常采用启发式方法，近似求解这一最优化问题. 这样得到的决策树是次最优（sub-optimal）的.

决策树学习的算法通常是一个递归地选择最优特征，并根据该特征对训练数据进行分割，使得对各个子数据集有一个最好的分类的过程. 这一过程对应着对特征空间的划分，也对应着决策树的构建. 开始，构建根结点，将所有训练数据

都放在根结点. 选择一个最优特征，按照这一特征将训练数据集分割成子集，使得各个子集有一个在当前条件下最好的分类. 如果这些子集已经能够被基本正确分类，那么构建叶结点，并将这些子集分到所对应的叶结点中去；如果还有子集不能被基本正确分类，那么就对这些子集选择新的最优特征，继续对其进行分割，构建相应的结点. 如此递归地进行下去，直至所有训练数据子集被基本正确分类，或者没有合适的特征为止. 最后每个子集都被分到叶结点上，即都有了明确的类. 这就生成了一棵决策树.

以上方法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却未必有很好的分类能力，即可能发生过拟合现象. 我们需要对已生成的树自下而上进行剪枝，将树变得更简单，从而使它具有更好的泛化能力. 具体地，就是去掉过于细分的叶结点，使其回退到父结点，甚至更高的结点，然后将父结点或更高的结点改为新的叶结点.

如果特征数量很多，也可以在决策树学习开始的时候，对特征进行选择，只留下对训练数据有足够分类能力的特征.

可以看出，决策树学习算法包含特征选择、决策树的生成与决策树的剪枝过程. 由于决策树表示一个条件概率分布，所以深浅不同的决策树对应着不同复杂度的概率模型. 决策树的生成对应于模型的局部选择，决策树的剪枝对应于模型的全局选择. 决策树的生成只考虑局部最优，相对地，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 与 CART，下面结合这些算法分别叙述决策树学习的特征选择、决策树的生成和剪枝过程.

5.2 特 征 选 择

5.2.1 特征选择问题

特征选择在于选取对训练数据具有分类能力的特征. 这样可以提高决策树学习的效率. 如果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别，则称这个特征是没有分类能力的. 经验上扔掉这样的特征对决策树学习的精度影响不大. 通常特征选择的准则是信息增益或信息增益比.

首先通过一个例子来说明特征选择问题.

例 5.1[①] 表 5.1 是一个由 15 个样本组成的贷款申请训练数据. 数据包括贷款申请人的 4 个特征（属性）：第 1 个特征是年龄，有 3 个可能值：青年，中年，老年；第 2 个特征是有工作，有 2 个可能值：是，否；第 3 个特征是有自己的房子，有 2 个可能值：是，否；第 4 个特征是信贷情况，有 3 个可能值：非常好，好，一般. 表的最后一列是类别，是否同意贷款，取 2 个值：是，否.

① 此例取自参考文献[5].

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

希望通过所给的训练数据学习一个贷款申请的决策树，用以对未来的贷款申请进行分类，即当新的客户提出贷款申请时，根据申请人的特征利用决策树决定是否批准贷款申请.

特征选择是决定用哪个特征来划分特征空间.

图 5.3 表示从表 5.1 数据学习到的两个可能的决策树，分别由两个不同特征的根结点构成. 图 5.3（a）所示的根结点的特征是年龄，有 3 个取值，对应于不同的取值有不同的子结点. 图 5.3（b）所示的根结点的特征是有工作，有 2 个取值，对应于不同的取值有不同的子结点. 两个决策树都可以从此延续下去. 问题是：究竟选择哪个特征更好些？这就要求确定选择特征的准则. 直观上，如果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成子集，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（information gain）就能够很好地表示这一直观的准则.

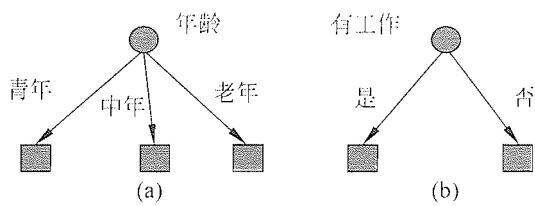

图 5.3 不同特征决定的不同决策树

5.2.2 信息增益

为了便于说明，先给出熵与条件熵的定义.

在信息论与概率统计中，熵（entropy）是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

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

则随机变量 X 的熵定义为

$$H(X)=-\sum_{i=1}^{n} p_i \log p_i \tag{5.1}$$

在式 (5.1) 中，若 $p_i=0$，则定义 $0\log 0=0$. 通常，式 (5.1) 中的对数以 2 为底或以 e 为底（自然对数），这时熵的单位分别称作比特（bit）或纳特（nat）. 由定义可知，熵只依赖于 X 的分布，而与 X 的取值无关，所以也可将 X 的熵记作 $H(p)$，即

$$H(p)=-\sum_{i=1}^{n} p_i \log p_i \tag{5.2}$$

熵越大，随机变量的不确定性就越大. 从定义可验证

$$0 \leqslant H(p) \leqslant \log n \tag{5.3}$$

当随机变量只取两个值，例如 1，0 时，即 X 的分布为

$$P(X=1)=p, \quad P(X=0)=1-p, \quad 0 \leqslant p \leqslant 1$$

熵为

$$H(p)=-p\log_2 p-(1-p)\log_2(1-p) \tag{5.4}$$

这时，熵 $H(p)$ 随概率 p 变化的曲线如图 5.4 所示（单位为比特）.

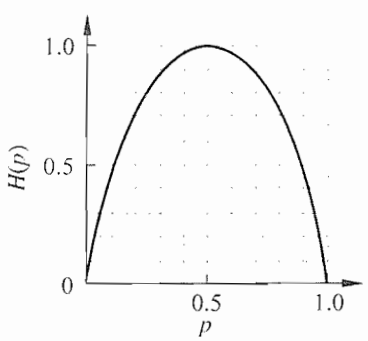

图 5.4 分布为贝努利分布时熵与概率的关系

当 $p=0$ 或 $p=1$ 时 $H(p)=0$，随机变量完全没有不确定性. 当 $p=0.5$ 时，$H(p)=1$，熵取值最大，随机变量不确定性最大.

设有随机变量(X,Y)，其联合概率分布为

$$P(X=x_i,Y=y_j)=p_{ij}\,,\quad i=1,2,\cdots,n;\quad j=1,2,\cdots,m$$

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. 随机变量X给定的条件下随机变量Y的条件熵（conditional entropy）$H(Y|X)$，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

$$H(Y|X)=\sum_{i=1}^{n}p_iH(Y|X=x_i) \tag{5.5}$$

这里，$p_i=P(X=x_i)$，$i=1,2,\cdots,n$.

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（empirical entropy）和经验条件熵（empirical conditional entropy）. 此时，如果有0概率，令$0\log 0=0$.

信息增益（information gain）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.

定义5.2（信息增益） 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

$$g(D,A)=H(D)-H(D|A) \tag{5.6}$$

一般地，熵$H(Y)$与条件熵$H(Y|X)$之差称为互信息（mutual information）. 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.

决策树学习应用信息增益准则选择特征. 给定训练数据集D和特征A，经验熵$H(D)$表示对数据集D进行分类的不确定性. 而经验条件熵$H(D|A)$表示在特征A给定的条件下对数据集D进行分类的不确定性. 那么它们的差，即信息增益，就表示由于特征A而使得对数据集D的分类的不确定性减少的程度. 显然，对于数据集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.

根据信息增益准则的特征选择方法是：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.

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. 于是信息增益的算法如下：

算法5.1（信息增益的算法）

输入：训练数据集D和特征A；

输出：特征 A 对训练数据集 D 的信息增益 $g(D,A)$.

（1）计算数据集 D 的经验熵 $H(D)$

$$H(D)=-\sum_{k=1}^{K}\frac{|C_k|}{|D|}\log_2\frac{|C_k|}{|D|} \tag{5.7}$$

（2）计算特征 A 对数据集 D 的经验条件熵 $H(D|A)$

$$H(D|A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}\log_2\frac{|D_{ik}|}{|D_i|} \tag{5.8}$$

（3）计算信息增益

$$g(D,A)=H(D)-H(D|A) \tag{5.9}$$ ■

例 5.2 对表 5.1 所给的训练数据集 D，根据信息增益准则选择最优特征.

解 首先计算经验熵 $H(D)$.

$$H(D)=-\frac{9}{15}\log_2\frac{9}{15}-\frac{6}{15}\log_2\frac{6}{15}=0.971$$

然后计算各特征对数据集 D 的信息增益，分别以 A_1，A_2，A_3、A_4 表示年龄、有工作、有自己的房子和信贷情况 4 个特征，则

（1）

$$\begin{aligned}g(D,A_1)&=H(D)-\left[\frac{5}{15}H(D_1)+\frac{5}{15}H(D_2)+\frac{5}{15}H(D_3)\right]\\&=0.971-\left[\frac{5}{15}\left(-\frac{2}{5}\log_2\frac{2}{5}-\frac{3}{5}\log_2\frac{3}{5}\right)\right.\\&\quad\left.+\frac{5}{15}\left(-\frac{3}{5}\log_2\frac{3}{5}-\frac{2}{5}\log_2\frac{2}{5}\right)+\frac{5}{15}\left(-\frac{4}{5}\log_2\frac{4}{5}-\frac{1}{5}\log_2\frac{1}{5}\right)\right]\\&=0.971-0.888=0.083\end{aligned}$$

这里 D_1，D_2，D_3 分别是 D 中 A_1（年龄）取值为青年、中年和老年的样本子集. 类似地，

（2）

$$\begin{aligned}g(D,A_2)&=H(D)-\left[\frac{5}{15}H(D_1)+\frac{10}{15}H(D_2)\right]\\&=0.971-\left[\frac{5}{15}\times 0+\frac{10}{15}\left(-\frac{4}{10}\log_2\frac{4}{10}-\frac{6}{10}\log_2\frac{6}{10}\right)\right]=0.324\end{aligned}$$

（3）

$$\begin{aligned}g(D,A_3)&=0.971-\left[\frac{6}{15}\times 0+\frac{9}{15}\left(-\frac{3}{9}\log_2\frac{3}{9}-\frac{6}{9}\log_2\frac{6}{9}\right)\right]\\&=0.971-0.551=0.420\end{aligned}$$

（4）

$$g(D,A_4)=0.971-0.608=0.363$$

最后，比较各特征的信息增益值．由于特征 A_3（有自己的房子）的信息增益值最大，所以选择特征 A_3 作为最优特征． ■

5.2.3 信息增益比

信息增益值的大小是相对于训练数据集而言的，并没有绝对意义．在分类问题困难时，也就是说在训练数据集的经验熵大的时候，信息增益值会偏大．反之，信息增益值会偏小．使用信息增益比（information gain ratio）可以对这一问题进行校正．这是特征选择的另一准则．

定义 5.3（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 的经验熵 $H(D)$ 之比：

$$g_R(D,A)=\frac{g(D,A)}{H(D)} \tag{5.10}$$

5.3 决策树的生成

本节将介绍决策树学习的生成算法．首先介绍 ID3 的生成算法，然后再介绍 C4.5 中的生成算法．这些都是决策树学习的经典算法．

5.3.1 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，递归地构建决策树．具体方法是：从根结点（root node）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．

算法 5.2（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 $A=\varnothing$，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，递归地调用步 (1) ～步 (5)，得到子树 T_i，返回 T_i. ■

例 5.3 对表 5.1 的训练数据集，利用 ID3 算法建立决策树.

解 利用例 5.2 的结果，由于特征 A_3（有自己的房子）的信息增益值最大，所以选择特征 A_3 作为根结点的特征. 它将训练数据集 D 划分为两个子集 D_1（A_3 取值为“是”）和 D_2（A_3 取值为“否”）. 由于 D_1 只有同一类的样本点，所以它成为一个叶结点，结点的类标记为“是”.

对 D_2 则需从特征 A_1（年龄），A_2（有工作）和 A_4（信贷情况）中选择新的特征，计算各个特征的信息增益：

$$g(D_2, A_1) = H(D_2) - H(D_2 \mid A_1) = 0.918 - 0.667 = 0.251$$

$$g(D_2, A_2) = H(D_2) - H(D_2 \mid A_2) = 0.918$$

$$g(D_2, A_4) = H(D_2) - H(D_2 \mid A_4) = 0.474$$

选择信息增益最大的特征 A_2（有工作）作为结点的特征. 由于 A_2 有两个可能取值，从这一结点引出两个子结点：一个对应“是”（有工作）的子结点，包含 3 个样本，它们属于同一类，所以这是一个叶结点，类标记为“是”；另一个是对应“否”（无工作）的子结点，包含 6 个样本，它们也属于同一类，所以这也是一个叶结点，类标记为“否”.

这样生成一个如图 5.5 所示的决策树. 该决策树只用了两个特征（有两个内部结点）. ■

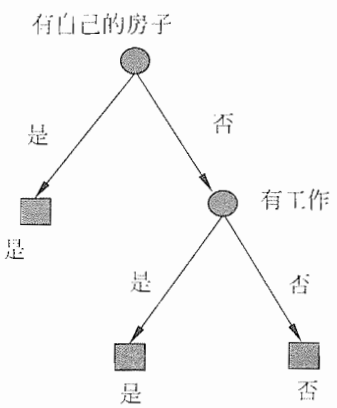

图 5.5 决策树的生成

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.

5.3.2 C4.5 的生成算法

C4.5 算法与 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.

算法 5.3（C4.5 的生成算法）

输入：训练数据集 D，特征集 A，阈值 ε；

输出：决策树 T.

（1）如果 D 中所有实例属于同一类 C_k，则置 T 为单结点树，并将 C_k 作为该结点的类，返回 T；

（2）如果 $A=\varnothing$，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，返回 T；

（3）否则，按式(5.10)计算 A 中各特征对 D 的信息增益比，选择信息增益比最大的特征 A_g；

（4）如果 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，返回 T；

（5）否则，对 A_g 的每一可能值 a_i，依 $A_g=a_i$ 将 D 分割为子集若干非空 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）对结点 i，以 D_i 为训练集，以 $A-\{A_g\}$ 为特征集，递归地调用步(1)～步(5)，得到子树 T_i，返回 T_i. ■

5.4 决策树的剪枝

决策树生成算法递归地产生决策树，直到不能继续下去为止．这样产生的树往往对训练数据的分类很准确，但对未知的测试数据的分类却没有那么准确，即出现过拟合现象．过拟合的原因在于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．解决这个问题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.

在决策树学习中将已生成的树进行简化的过程称为剪枝（pruning）．具体地，剪枝从已生成的树上裁掉一些子树或叶结点，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.

本节介绍一种简单的决策树学习的剪枝算法.

决策树的剪枝往往通过极小化决策树整体的损失函数（loss function）或代价函数（cost function）来实现．设树 T 的叶结点个数为 $|T|$，t 是树 T 的叶结点，该

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，$\alpha \geqslant 0$ 为参数，则决策树学习的损失函数可以定义为

$$C_\alpha(T)=\sum_{t=1}^{|T|} N_t H_t(T)+\alpha|T| \tag{5.11}$$

其中经验熵为

$$H_t(T)=-\sum_k \frac{N_{tk}}{N_t}\log\frac{N_{tk}}{N_t} \tag{5.12}$$

在损失函数中，将式 (5.11) 右端的第 1 项记作

$$C(T)=\sum_{t=1}^{|T|} N_t H_t(T)=-\sum_{t=1}^{|T|}\sum_{k=1}^{K} N_{tk}\log\frac{N_{tk}}{N_t} \tag{5.13}$$

这时有

$$C_\alpha(T)=C(T)+\alpha|T| \tag{5.14}$$

式(5.14)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 $\alpha \geqslant 0$ 控制两者之间的影响．较大的 α 促使选择较简单的模型（树），较小的 α 促使选择较复杂的模型（树）．$\alpha=0$ 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．

剪枝，就是当 α 确定时，选择损失函数最小的模型，即损失函数最小的子树．当 α 值确定时，子树越大，往往与训练数据的拟合越好，但是模型的复杂度就越高；相反，子树越小，模型的复杂度就越低，但是往往与训练数据的拟合不好．损失函数正好表示了对两者的平衡．

可以看出，决策树生成只考虑了通过提高信息增益（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．

式 (5.11) 或式 (5.14) 定义的损失函数的极小化等价于正则化的极大似然估计．所以，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．

图 5.6 是决策树剪枝过程的示意图．下面介绍剪枝算法．

算法 5.4（树的剪枝算法）

输入：生成算法产生的整个树 T，参数 α；

输出：修剪后的子树 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．

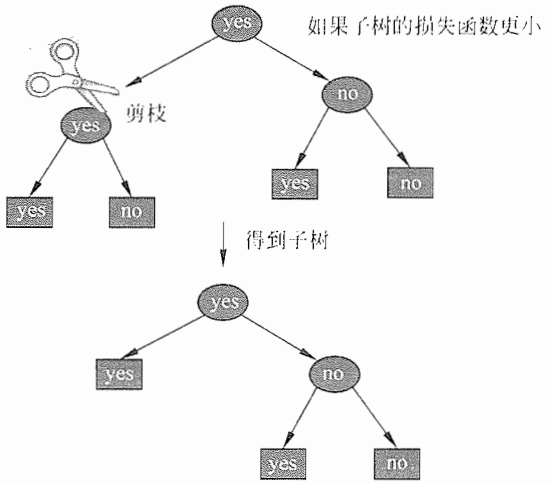

图 5.6 决策树的剪枝

设一组叶结点回缩到其父结点之前与之后的整体树分别为 T_B 与 T_A，其对应的损失函数值分别是 $C_\alpha(T_B)$ 与 $C_\alpha(T_A)$，如果

$$C_\alpha(T_A) \leqslant C_\alpha(T_B) \tag{5.15}$$

则进行剪枝，即将父结点变为新的叶结点.

(3) 返回 (2)，直至不能继续为止，得到损失函数最小的子树 T_α. ■

注意，式 (5.15) 只需考虑两个树的损失函数的差，其计算可以在局部进行. 所以，决策树的剪枝算法可以由一种动态规划的算法实现. 类似的动态规划算法可参见文献[10].

5.5 CART 算法

分类与回归树（classification and regression tree，CART）模型由 Breiman 等人在 1984 年提出，是应用广泛的决策树学习方法. CART 同样由特征选择、树的生成及剪枝组成，既可以用于分类也可以用于回归. 以下将用于分类与回归的树统称为决策树.

CART 是在给定输入随机变量 X 条件下输出随机变量 Y 的条件概率分布的学习方法. CART 假设决策树是二叉树，内部结点特征的取值为“是”和“否”，左分支是取值为“是”的分支，右分支是取值为“否”的分支. 这样的决策树等价于递归地二分每个特征，将输入空间即特征空间划分为有限个单元，并在这些单元上确定预测的概率分布，也就是在输入给定的条件下输出的条件概率分布.

CART 算法由以下两步组成：

(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，这时用损失函数最小作为剪枝的标准.

5.5.1 CART生成

决策树的生成就是递归地构建二叉决策树的过程.对回归树用平方误差最小化准则，对分类树用基尼指数（Gini index）最小化准则，进行特征选择，生成二叉树.

1. 回归树的生成

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

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

考虑如何生成回归树.

一个回归树对应着输入空间（即特征空间）的一个划分以及在划分的单元上的输出值.假设已将输入空间划分为 M 个单元 $R_1,R_2,\cdots,R_M$，并且在每个单元 R_m 上有一个固定的输出值 c_m，于是回归树模型可表示为

$$f(x)=\sum_{m=1}^{M}c_m I(x\in R_m) \tag{5.16}$$

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，用平方误差最小的准则求解每个单元上的最优输出值.易知，单元 R_m 上的 c_m 的最优值 $\hat{c}_m$ 是 R_m 上的所有输入实例 x_i 对应的输出 y_i 的均值，即

$$\hat{c}_m=\text{ave}(y_i\mid x_i\in R_m) \tag{5.17}$$

问题是怎样对输入空间进行划分．这里采用启发式的方法，选择第 j 个变量 $x^{(j)}$ 和它取的值 s，作为切分变量（splitting variable）和切分点（splitting point），并定义两个区域：

$$R_1(j,s)=\{x\mid x^{(j)}\leqslant s\}\quad 和\quad R_2(j,s)=\{x\mid x^{(j)}>s\} \tag{5.18}$$

然后寻找最优切分变量 j 和最优切分点 s．具体地，求解

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right] \tag{5.19}$$

对固定输入变量 j 可以找到最优切分点 s.

$$\hat{c}_1=\text{ave}(y_i\mid x_i\in R_1(j,s))\quad 和\quad \hat{c}_2=\text{ave}(y_i\mid x_i\in R_2(j,s)) \tag{5.20}$$

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)．依此将输入空间

划分为两个区域. 接着，对每个区域重复上述划分过程，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(least squares regression tree)，现将算法叙述如下：

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：

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

2. 分类树的生成

分类树用基尼指数选择最优特征，同时决定该特征的最优二值切分点.

定义 5.4（基尼指数） 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，则概率分布的基尼指数定义为

$$\text{Gini}(p)=\sum_{k=1}^{K}p_k(1-p_k)=1-\sum_{k=1}^{K}p_k^2 \tag{5.22}$$

对于二类分类问题，若样本点属于第 1 个类的概率是 p，则概率分布的基尼指数为

$$\text{Gini}(p)=2p(1-p) \tag{5.23}$$

对于给定的样本集合 D，其基尼指数为

$$\text{Gini}(D)=1-\sum_{k=1}^{K}\left(\frac{|C_k|}{|D|}\right)^2 \tag{5.24}$$

这里，C_k 是 D 中属于第 k 类的样本子集，K 是类的个数.

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_2 两部分，即

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

则在特征 A 的条件下，集合 D 的基尼指数定义为

$$\mathrm{Gini}(D,A)=\frac{|D_1|}{|D|}\mathrm{Gini}(D_1)+\frac{|D_2|}{|D|}\mathrm{Gini}(D_2) \tag{5.25}$$

基尼指数 $\mathrm{Gini}(D)$ 表示集合 D 的不确定性，基尼指数 $\mathrm{Gini}(D,A)$ 表示经 $A=a$ 分割后集合 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，这一点与熵相似.

图 5.7 显示二类分类问题中基尼指数 $\mathrm{Gini}(p)$、熵（单位比特）之半 $\frac{1}{2}H(p)$ 和分类误差率的关系. 横坐标表示概率 p，纵坐标表示损失. 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.

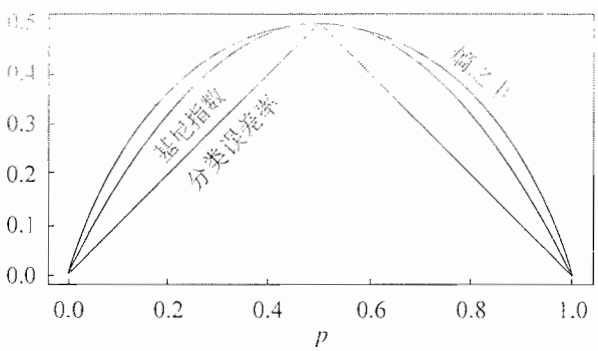

图 5.7 二类分类中基尼指数、熵之半和分类误差率的关系

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.

例 5.4 根据表 5.1 所给训练数据集，应用 CART 算法生成决策树.

解 首先计算各特征的基尼指数，选择最优特征以及其最优切分点. 仍采用例 5.2 的记号，分别以 A_1，A_2，A_3，A_4 表示年龄、有工作、有自己的房子和信贷情况 4 个特征，并以 1，2，3 表示年龄的值为青年、中年和老年，以 1，2 表示有工作和有自己的房子的值为是和否，以 1，2，3 表示信贷情况的值为非常好、好和一般.

求特征 A_1 的基尼指数:

$$\mathrm{Gini}(D, A_1=1)=\frac{5}{15}\left(2\times\frac{2}{5}\times\left(1-\frac{2}{5}\right)\right)+\frac{10}{15}\left(2\times\frac{7}{10}\times\left(1-\frac{7}{10}\right)\right)=0.44$$

$$\mathrm{Gini}(D, A_1=2)=0.48$$

$$\mathrm{Gini}(D, A_1=3)=0.44$$

由于 $\mathrm{Gini}(D, A_1=1)$ 和 $\mathrm{Gini}(D, A_1=3)$ 相等，且最小，所以 $A_1=1$ 和 $A_1=3$ 都可以选作 A_1 的最优切分点.

求特征 A_2 和 A_3 的基尼指数:

$$\mathrm{Gini}(D, A_2=1)=0.32$$

$$\mathrm{Gini}(D, A_3=1)=0.27$$

由于 A_2 和 A_3 只有一个切分点，所以它们就是最优切分点.

求特征 A_4 的基尼指数:

$$\mathrm{Gini}(D, A_4=1)=0.36$$

$$\mathrm{Gini}(D, A_4=2)=0.47$$

$$\mathrm{Gini}(D, A_4=3)=0.32$$

$\mathrm{Gini}(D, A_4=3)$ 最小，所以 $A_4=3$ 为 A_4 的最优切分点.

在 A_1，A_2，A_3，A_4 几个特征中，$\mathrm{Gini}(D, A_3=1)=0.27$ 最小，所以选择特征 A_3 为最优特征，$A_3=1$ 为其最优切分点. 于是根结点生成两个子结点，一个是叶结点. 对另一个结点继续使用以上方法在 A_1，A_2，A_4 中选择最优特征及其最优切分点，结果是 $A_2=1$. 依此计算得知，所得结点都是叶结点. ■

对于本问题，按照 CART 算法所生成的决策树与按照 ID3 算法所生成的决策树完全一致.

5.5.2 CART剪枝

CART剪枝算法从“完全生长”的决策树的底端剪去一些子树，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．CART剪枝算法由两步组成：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的根结点，形成一个子树序列$\{T_0,T_1,\cdots,T_n\}$；然后通过交叉验证法在独立的验证数据集上对子树序列进行测试，从中选择最优子树．

1. 剪枝，形成一个子树序列

在剪枝过程中，计算子树的损失函数：

$$C_\alpha(T)=C(T)+\alpha|T| \tag{5.26}$$

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

对固定的α，一定存在使损失函数$C_\alpha(T)$最小的子树，将其表示为T_α．T_α在损失函数$C_\alpha(T)$最小的意义下是最优的．容易验证这样的最优子树是唯一的．当α大的时候，最优子树T_α偏小；当α小的时候，最优子树T_α偏大．极端情况，当$\alpha=0$时，整体树是最优的．当$\alpha\to\infty$时，根结点组成的单结点树是最优的．

Breiman等人证明：可以用递归的方法对树进行剪枝．将α从小增大，$0=\alpha_0<\alpha_1<\cdots<\alpha_n<+\infty$，产生一系列的区间$[\alpha_i,\alpha_{i+1}),i=0,1,\cdots,n$；剪枝得到的子树序列对应着区间$\alpha\in[\alpha_i,\alpha_{i+1})$，$i=0,1,\cdots,n$的最优子树序列$\{T_0,T_1,\cdots,T_n\}$，序列中的子树是嵌套的．

具体地，从整体树T_0开始剪枝．对T_0的任意内部结点t，以t为单结点树的损失函数是

$$C_\alpha(t)=C(t)+\alpha \tag{5.27}$$

以t为根结点的子树T_t的损失函数是

$$C_\alpha(T_t)=C(T_t)+\alpha|T_t| \tag{5.28}$$

当$\alpha=0$及α充分小时，有不等式

$$C_\alpha(T_t)<C_\alpha(t) \tag{5.29}$$

当α增大时，在某一α有

$$C_\alpha(T_t)=C_\alpha(t) \tag{5.30}$$

当α再增大时，不等式(5.29)反向．只要$\alpha=\dfrac{C(t)-C(T_t)}{|T_t|-1}$，$T_t$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 中每一内部结点 t，计算

$$g(t)=\frac{C(t)-C(T_t)}{|T_t|-1} \tag{5.31}$$

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 α_1．T_1 为区间 $[\alpha_1,\alpha_2)$ 的最优子树．

如此剪枝下去，直至得到根结点．在这一过程中，不断地增加 α 的值，产生新的区间．

2．在剪枝得到的子树序列 $T_0,T_1,\cdots,T_n$ 中通过交叉验证选取最优子树 T_α

具体地，利用独立的验证数据集，测试子树序列 $T_0,T_1,\cdots,T_n$ 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 $T_1,T_2,\cdots,T_n$ 都对应于一个参数 $\alpha_1,\alpha_2,\cdots,\alpha_n$．所以，当最优子树 T_k 确定时，对应的 α_k 也确定了，即得到最优决策树 T_α．

现在写出 CART 剪枝算法．

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． ■

本 章 概 要

1．分类决策树模型是表示基于特征对实例进行分类的树形结构．决策树可以转换成一个 if-then 规则的集合，也可以看作是定义在特征空间划分上的类的条

件概率分布.

2．决策树学习旨在构建一个与训练数据拟合很好，并且复杂度小的决策树．因为从可能的决策树中直接选取最优决策树是 NP 完全问题．现实中采用启发式方法学习次优的决策树．

决策树学习算法包括 3 部分：特征选择、树的生成和树的剪枝．常用的算法有 ID3、C4.5 和 CART.

3．特征选择的目的在于选取对训练数据能够分类的特征．特征选择的关键是其准则．常用的准则如下：

（1）样本集合 D 对特征 A 的信息增益（ID3）

$$g(D,A)=H(D)-H(D\mid A)$$

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

$$H(D\mid A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)$$

其中，$H(D)$ 是数据集 D 的熵，$H(D_i)$ 是数据集 D_i 的熵，$H(D\mid A)$ 是数据集 D 对特征 A 的条件熵．D_i 是 D 中特征 A 取第 i 个值的样本子集，C_k 是 D 中属于第 k 类的样本子集．n 是特征 A 取值的个数，K 是类的个数．

（2）样本集合 D 对特征 A 的信息增益比（C4.5）

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

其中，$g(D,A)$ 是信息增益，$H(D)$ 是数据集 D 的熵．

（3）样本集合 D 的基尼指数（CART）

$$\mathrm{Gini}(D)=1-\sum_{k=1}^{K}\left(\frac{|C_k|}{|D|}\right)^2$$

特征 A 条件下集合 D 的基尼指数：

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

4．决策树的生成．通常使用信息增益最大、信息增益比最大或基尼指数最小作为特征选择的准则．决策树的生成往往通过计算信息增益或其他指标，从根结点开始，递归地产生决策树．这相当于用信息增益或其他准则不断地选取局部最优的特征，或将训练集分割为能够基本正确分类的子集．

5．决策树的剪枝．由于生成的决策树存在过拟合问题，需要对它进行剪枝，以简化学到的决策树．决策树的剪枝，往往从已生成的树上剪掉一些叶结点或叶结点以上的子树，并将其父结点或根结点作为新的叶结点，从而简化生成的决策树．

继续阅读

介绍决策树学习方法的文献很多，关于 ID3 可见文献[1]，C4.5 可见文献[2]，CART 可见文献[3, 4]. 决策树学习一般性介绍可见文献[5～7]. 与决策树类似的分类方法还有决策列表（decision list）. 决策列表与决策树可以相互转换[8]，决策列表的学习方法可参见文献[9].

习　题

5.1　根据表 5.1 所给的训练数据集，利用信息增益比（C4.5 算法）生成决策树.

5.2　已知如表 5.2 所示的训练数据，试用平方误差损失准则生成一个二叉回归树.

表 5.2　训练数据表

x_i	1	2	3	4	5	6	7	8	9	10
y_i	4.50	4.75	4.91	5.34	5.80	7.05	7.90	8.23	8.70	9.00

5.3　证明 CART 剪枝算法中，当 α 确定时，存在唯一的最小子树 T_α 使损失函数 $C_\alpha(T)$ 最小.

5.4　证明 CART 剪枝算法中求出的子树序列 $\{T_0, T_1, \cdots, T_n\}$ 分别是区间 $\alpha \in [\alpha_i, \alpha_{i+1})$ 的最优子树 T_α，这里 $i = 0, 1, \cdots, n$，$0 = \alpha_0 < \alpha_1 < \cdots < \alpha_n < +\infty$.

参 考 文 献

[1] Quinlan JR. Induction of decision trees. Machine Learning, 1986, 1(1): 81–106

[2] Quinlan JR. C4. 5: Programs for Machine Learning. Morgan Kaufmann, 1992

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

[4] Ripley B. Pattern Recognition and Neural Networks. Cambridge University Press, 1996

[5] Liu B. Web Data Mining: Exploring Hyperlinks, Contents and Usage Data. Springer-Verlag, 2006

[6] Hyafil L, Rivest RL. Constructing Optimal Binary Decision Trees is NP-complete. Information Processing Letters, 1976, 5(1): 15–17

[7] Hastie T, Tibshirani R, Friedman JH. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. New York: Springer-Verlag, 2001

[8] Yamanishi K. A learning criterion for stochastic rules. Machine Learning, 1992

[9] Li H, Yamanishi K. Text classification using ESC-based stochastic decision lists. Information Processing & Management, 2002, 38(3): 343–361

[10] Li H, Abe N. Generalizing case frames using a thesaurus and the MDL principle. Computational Linguistics, 1998, 24(2): 217–244

第7章　支持向量机

支持向量机（support vector machines，SVM）是一种二类分类模型．它的基本模型是定义在特征空间上的间隔最大的线性分类器，间隔最大使它有别于感知机；支持向量机还包括核技巧，这使它成为实质上的非线性分类器．支持向量机的学习策略就是间隔最大化，可形式化为一个求解凸二次规划（convex quadratic programming）的问题，也等价于正则化的合页损失函数的最小化问题．支持向量机的学习算法是求解凸二次规划的最优化算法．

支持向量机学习方法包含构建由简至繁的模型：线性可分支持向量机（linear support vector machine in linearly separable case）、线性支持向量机（linear support vector machine）及非线性支持向量机（non-linear support vector machine）．简单模型是复杂模型的基础，也是复杂模型的特殊情况．当训练数据线性可分时，通过硬间隔最大化（hard margin maximization），学习一个线性的分类器，即线性可分支持向量机，又称为硬间隔支持向量机；当训练数据近似线性可分时，通过软间隔最大化（soft margin maximization），也学习一个线性的分类器，即线性支持向量机，又称为软间隔支持向量机；当训练数据线性不可分时，通过使用核技巧（kernel trick）及软间隔最大化，学习非线性支持向量机．

当输入空间为欧氏空间或离散集合、特征空间为希尔伯特空间时，核函数（kernel function）表示将输入从输入空间映射到特征空间得到的特征向量之间的内积．通过使用核函数可以学习非线性支持向量机，等价于隐式地在高维的特征空间中学习线性支持向量机．这样的方法称为核技巧．核方法（kernel method）是比支持向量机更为一般的机器学习方法．

Cortes 与 Vapnik 提出线性支持向量机，Boser、Guyon 与 Vapnik 又引入核技巧，提出非线性支持向量机．

本章按照上述思路介绍 3 类支持向量机、核函数及一种快速学习算法——序列最小最优化算法（SMO）．

7.1　线性可分支持向量机与硬间隔最大化

7.1.1　线性可分支持向量机

考虑一个二类分类问题．假设输入空间与特征空间为两个不同的空间．输入空间为欧氏空间或离散集合，特征空间为欧氏空间或希尔伯特空间．线性可分支

持向量机、线性支持向量机假设这两个空间的元素一一对应，并将输入空间中的输入映射为特征空间中的特征向量. 非线性支持向量机利用一个从输入空间到特征空间的非线性映射将输入映射为特征向量. 所以，输入都由输入空间转换到特征空间，支持向量机的学习是在特征空间进行的.

假设给定一个特征空间上的训练数据集

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

其中，$x_i\in\mathcal{X}=\mathbf{R}^n$，$y_i\in\mathcal{Y}=\{+1,-1\}$，$i=1,2,\cdots,N$，$x_i$ 为第 i 个特征向量，也称为实例，y_i 为 x_i 的类标记，当 $y_i=+1$ 时，称 x_i 为正例；当 $y_i=-1$ 时，称 x_i 为负例，(x_i,y_i) 称为样本点. 再假设训练数据集是线性可分的（见定义 2.2）.

学习的目标是在特征空间中找到一个分离超平面，能将实例分到不同的类. 分离超平面对应于方程 $w\cdot x+b=0$，它由法向量 w 和截距 b 决定，可用 (w,b) 来表示. 分离超平面将特征空间划分为两部分，一部分是正类，一部分是负类. 法向量指向的一侧为正类，另一侧为负类.

一般地，当训练数据集线性可分时，存在无穷个分离超平面可将两类数据正确分开. 感知机利用误分类最小的策略，求得分离超平面，不过这时的解有无穷多个. 线性可分支持向量机利用间隔最大化求最优分离超平面，这时，解是唯一的.

定义 7.1（线性可分支持向量机） 给定线性可分训练数据集，通过间隔最大化或等价地求解相应的凸二次规划问题学习得到的分离超平面为

$$w^*\cdot x+b^*=0 \tag{7.1}$$

以及相应的分类决策函数

$$f(x)=\operatorname{sign}(w^*\cdot x+b^*) \tag{7.2}$$

称为线性可分支持向量机.

考虑如图 7.1 所示的二维特征空间中的分类问题. 图中“○”表示正例，“×”表示负例. 训练数据集线性可分，这时有许多直线能将两类数据正确划分. 线性可分支持向量机对应着将两类数据正确划分并且间隔最大的直线，如图 7.1 所示.

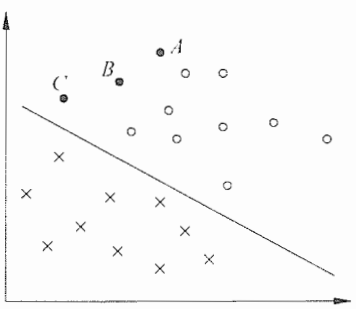

图 7.1 二类分类问题

间隔最大及相应的约束最优化问题将在下面叙述．这里先介绍函数间隔和几何间隔的概念．

7.1.2 函数间隔和几何间隔

在图 7.1 中，有 A，B，C 三个点，表示 3 个实例，均在分离超平面的正类一侧，预测它们的类．点 A 距分离超平面较远，若预测该点为正类，就比较确信预测是正确的；点 C 距分离超平面较近，若预测该点为正类就不那么确信；点 B 介于点 A 与 C 之间，预测其为正类的确信度也在 A 与 C 之间．

一般来说，一个点距离分离超平面的远近可以表示分类预测的确信程度．在超平面 $w\cdot x+b=0$ 确定的情况下，$|w\cdot x+b|$ 能够相对地表示点 x 距离超平面的远近．而 $w\cdot x+b$ 的符号与类标记 y 的符号是否一致能够表示分类是否正确．所以可用量 $y(w\cdot x+b)$ 来表示分类的正确性及确信度，这就是函数间隔（functional margin）的概念．

定义 7.2（函数间隔） 对于给定的训练数据集 T 和超平面 (w,b)，定义超平面 (w,b) 关于样本点 (x_i,y_i) 的函数间隔为

$$\hat{\gamma}_i=y_i(w\cdot x_i+b) \tag{7.3}$$

定义超平面 (w,b) 关于训练数据集 T 的函数间隔为超平面 (w,b) 关于 T 中所有样本点 (x_i,y_i) 的函数间隔之最小值，即

$$\hat{\gamma}=\min_{i=1,\cdots,N}\hat{\gamma}_i \tag{7.4}$$

函数间隔可以表示分类预测的正确性及确信度．但是选择分离超平面时，只有函数间隔还不够．因为只要成比例地改变 w 和 b，例如将它们改为 $2w$ 和 $2b$，超平面并没有改变，但函数间隔却成为原来的 2 倍．这一事实启示我们，可以对分离超平面的法向量 w 加某些约束，如规范化，$\|w\|=1$，使得间隔是确定的．这时函数间隔成为几何间隔（geometric margin）．

图 7.2 给出了超平面 (w,b) 及其法向量 w．点 A 表示某一实例 x_i，其类标记为 $y_i=+1$．点 A 与超平面 (w,b) 的距离由线段 AB 给出，记作 γ_i．

$$\gamma_i=\frac{w}{\|w\|}\cdot x_i+\frac{b}{\|w\|}$$

其中，$\|w\|$ 为 w 的 L_2 范数．这是点 A 在超平面正的一侧的情形．如果点 A 在超平面负的一侧，即 $y_i=-1$，那么点与超平面的距离为

$$\gamma_i=-\left(\frac{w}{\|w\|}\cdot x_i+\frac{b}{\|w\|}\right)$$

一般地，当样本点 (x_i, y_i) 被超平面 (w,b) 正确分类时，点 x_i 与超平面 (w,b) 的距离是

$$\gamma_i = y_i\left(\frac{w}{\|w\|}\cdot x_i + \frac{b}{\|w\|}\right)$$

由这一事实导出几何间隔的概念.

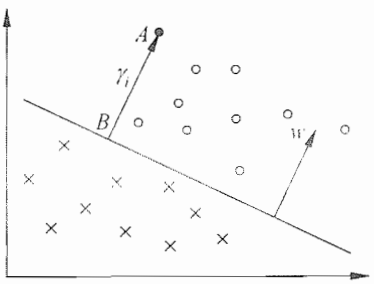

图 7.2 几何间隔

定义 7.3（几何间隔） 对于给定的训练数据集 T 和超平面 (w,b)，定义超平面 (w,b) 关于样本点 (x_i, y_i) 的几何间隔为

$$\gamma_i = y_i\left(\frac{w}{\|w\|}\cdot x_i + \frac{b}{\|w\|}\right) \tag{7.5}$$

定义超平面 (w,b) 关于训练数据集 T 的几何间隔为超平面 (w,b) 关于 T 中所有样本点 (x_i, y_i) 的几何间隔之最小值，即

$$\gamma = \min_{i=1,\cdots,N} \gamma_i \tag{7.6}$$

超平面 (w,b) 关于样本点 (x_i, y_i) 的几何间隔一般是实例点到超平面的带符号的距离（signed distance），当样本点被超平面正确分类时就是实例点到超平面的距离.

从函数间隔和几何间隔的定义（式(7.3) ～ 式(7.6)）可知，函数间隔和几何间隔有下面的关系：

$$\gamma_i = \frac{\hat{\gamma}_i}{\|w\|} \tag{7.7}$$

$$\gamma = \frac{\hat{\gamma}}{\|w\|} \tag{7.8}$$

如果 $\|w\|=1$，那么函数间隔和几何间隔相等. 如果超平面参数 w 和 b 成比例地改变（超平面没有改变），函数间隔也按此比例改变，而几何间隔不变.

7.1.3 间隔最大化

支持向量机学习的基本想法是求解能够正确划分训练数据集并且几何间隔最大的分离超平面. 对线性可分的训练数据集而言，线性可分分离超平面有无穷多个（等价于感知机），但是几何间隔最大的分离超平面是唯一的. 这里的间隔最大化又称为硬间隔最大化（与将要讨论的训练数据集近似线性可分时的软间隔最大化相对应）.

间隔最大化的直观解释是：对训练数据集找到几何间隔最大的超平面意味着以充分大的确信度对训练数据进行分类. 也就是说，不仅将正负实例点分开，而且对最难分的实例点（离超平面最近的点）也有足够大的确信度将它们分开. 这样的超平面应该对未知的新实例有很好的分类预测能力.

1. 最大间隔分离超平面

下面考虑如何求得一个几何间隔最大的分离超平面，即最大间隔分离超平面. 具体地，这个问题可以表示为下面的约束最优化问题：

$$\max_{w,b} \quad \gamma \tag{7.9}$$

$$\text{s.t.} \quad y_i\left(\frac{w}{\|w\|}\cdot x_i+\frac{b}{\|w\|}\right)\geqslant \gamma, \quad i=1,2,\cdots,N \tag{7.10}$$

即我们希望最大化超平面 (w,b) 关于训练数据集的几何间隔 γ，约束条件表示的是超平面 (w,b) 关于每个训练样本点的几何间隔至少是 γ.

考虑几何间隔和函数间隔的关系式 (7.8)，可将这个问题改写为

$$\max_{w,b} \quad \frac{\hat{\gamma}}{\|w\|} \tag{7.11}$$

$$\text{s.t.} \quad y_i(w\cdot x_i+b)\geqslant \hat{\gamma}, \quad i=1,2,\cdots,N \tag{7.12}$$

函数间隔 $\hat{\gamma}$ 的取值并不影响最优化问题的解. 事实上，假设将 w 和 b 按比例改变为 λw 和 λb，这时函数间隔成为 $\lambda\hat{\gamma}$. 函数间隔的这一改变对上面最优化问题的不等式约束没有影响，对目标函数的优化也没有影响，也就是说，它产生一个等价的最优化问题. 这样，就可以取 $\hat{\gamma}=1$. 将 $\hat{\gamma}=1$ 代入上面的最优化问题，注意到最大化 $\frac{1}{\|w\|}$ 和最小化 $\frac{1}{2}\|w\|^2$ 是等价的，于是就得到下面的线性可分支持向量机学习的最优化问题

$$\min_{w,b} \quad \frac{1}{2}\|w\|^2 \tag{7.13}$$

$$\text{s.t.} \quad y_i(w\cdot x_i+b)-1\geqslant 0, \quad i=1,2,\cdots,N \tag{7.14}$$

这是一个凸二次规划（convex quadratic programming）问题.

凸优化问题是指约束最优化问题

$$\min_{w} \quad f(w) \tag{7.15}$$

$$\text{s.t.} \quad g_i(w) \leqslant 0, \quad i=1,2,\cdots,k \tag{7.16}$$

$$h_i(w)=0, \quad i=1,2,\cdots,l \tag{7.17}$$

其中，目标函数 $f(w)$ 和约束函数 $g_i(w)$ 都是 $\mathbf{R}^n$ 上的连续可微的凸函数，约束函数 $h_i(w)$ 是 $\mathbf{R}^n$ 上的仿射函数[①].

当目标函数 $f(w)$ 是二次函数且约束函数 $g_i(w)$ 是仿射函数时，上述凸最优化问题成为凸二次规划问题.

如果求出了约束最优化问题 (7.13) ～ (7.14) 的解 w^*, b^*，那么就可以得到最大间隔分离超平面 $w^* \cdot x + b^* = 0$ 及分类决策函数 $f(x)=\text{sign}(w^* \cdot x + b^*)$，即线性可分支持向量机模型.

综上所述，就有下面的线性可分支持向量机的学习算法——最大间隔法（maximum margin method）.

算法 7.1（线性可分支持向量机学习算法——最大间隔法）

输入：线性可分训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中，$x_i \in \mathcal{X}=\mathbf{R}^n$，$y_i \in \mathcal{Y}=\{-1,+1\}$，$i=1,2,\cdots,N$；

输出：最大间隔分离超平面和分类决策函数.

（1）构造并求解约束最优化问题：

$$\min_{w,b} \quad \frac{1}{2}\|w\|^2$$

$$\text{s.t.} \quad y_i(w \cdot x_i + b) - 1 \geqslant 0, \quad i=1,2,\cdots,N$$

求得最优解 w^*, b^*.

（2）由此得到分离超平面：

$$w^* \cdot x + b^* = 0$$

分类决策函数

$$f(x)=\text{sign}(w^* \cdot x + b^*)$$

2. 最大间隔分离超平面的存在唯一性

线性可分训练数据集的最大间隔分离超平面是存在且唯一的.

定理 7.1（最大间隔分离超平面的存在唯一性） 若训练数据集 T 线性可分，则可将训练数据集中的样本点完全正确分开的最大间隔分离超平面存在且唯一.

① $f(x)$ 称为仿射函数，如果它满足 $f(x)=a \cdot x + b$，$a \in \mathbf{R}^n$，$b \in \mathbf{R}$，$x \in \mathbf{R}^n$.

证明 （1）存在性

由于训练数据集线性可分，所以算法 7.1 中的最优化问题 (7.13) ～ (7.14) 一定存在可行解. 又由于目标函数有下界，所以最优化问题 (7.13) ～ (7.14) 必有解，记作 (w^*,b^*). 由于训练数据集中既有正类点又有负类点，所以 $(w,b)=(0,b)$ 不是最优化的可行解，因而最优解 (w^*,b^*) 必满足 $w^*\neq 0$. 由此得知分离超平面的存在性.

（2）唯一性

首先证明最优化问题 (7.13) ～ (7.14) 解中 w^* 的唯一性. 假设问题 (7.13) ～ (7.14) 存在两个最优解 (w_1^*,b_1^*) 和 (w_2^*,b_2^*). 显然 $\|w_1^*\|=\|w_2^*\|=c$，其中 c 是一个常数. 令 $w=\dfrac{w_1^*+w_2^*}{2}$，$b=\dfrac{b_1^*+b_2^*}{2}$，易知 (w,b) 是问题 (7.13) ～ (7.14) 的可行解，从而有

$$c\leqslant\|w\|\leqslant\frac{1}{2}\|w_1^*\|+\frac{1}{2}\|w_2^*\|=c$$

上式表明，式中的不等号可变为等号，即 $\|w\|=\dfrac{1}{2}\|w_1^*\|+\dfrac{1}{2}\|w_2^*\|$，从而有 $w_1^*=\lambda w_2^*$，$|\lambda|=1$. 若 $\lambda=-1$，则 $w=0$，(w,b) 不是问题 (7.13) ～ (7.14) 的可行解，矛盾. 因此必有 $\lambda=1$，即

$$w_1^*=w_2^*$$

由此可以把两个最优解 (w_1^*,b_1^*) 和 (w_2^*,b_2^*) 分别写成 (w^*,b_1^*) 和 (w^*,b_2^*). 再证 $b_1^*=b_2^*$. 设 x_1' 和 x_2' 是集合 $\{x_i\mid y_i=+1\}$ 中分别对应于 (w^*,b_1^*) 和 (w^*,b_2^*) 使得问题的不等式等号成立的点，x_1'' 和 x_2'' 是集合 $\{x_i\mid y_i=-1\}$ 中分别对应于 (w^*,b_1^*) 和 (w^*,b_2^*) 使得问题的不等式等号成立的点，则由 $b_1^*=-\dfrac{1}{2}(w^*\cdot x_1'+w^*\cdot x_1'')$，$b_2^*=-\dfrac{1}{2}(w^*\cdot x_2'+w^*\cdot x_2'')$，得

$$b_1^*-b_2^*=-\frac{1}{2}[(w^*\cdot(x_1'-x_2')+w^*\cdot(x_1''-x_2'')]$$

又因为

$$w^*\cdot x_2'+b_1^*\geqslant 1=w^*\cdot x_1'+b_1^*$$
$$w^*\cdot x_1'+b_2^*\geqslant 1=w^*\cdot x_2'+b_2^*$$

所以，$w^*\cdot(x_1'-x_2')=0$. 同理有 $w^*\cdot(x_1''-x_2'')=0$. 因此，

$$b_1^*-b_2^*=0$$

由 $w_1^*=w_2^*$ 和 $b_1^*=b_2^*$ 可知，两个最优解 (w_1^*,b_1^*) 和 (w_2^*,b_2^*) 是相同的，解的唯一性得证.

由问题 (7.13) ～ (7.14) 解的唯一性即得分离超平面是唯一的.

（3）分离超平面能将训练数据集中的两类点完全正确地分开.

由解满足问题的约束条件即可得知. ■

3. 支持向量和间隔边界

在线性可分情况下，训练数据集的样本点中与分离超平面距离最近的样本点的实例称为支持向量（support vector）. 支持向量是使约束条件式 (7.14) 等号成立的点，即

$$y_i(w\cdot x_i+b)-1=0$$

对 $y_i=+1$ 的正例点，支持向量在超平面

$$H_1: w\cdot x+b=1$$

上，对 $y_i=-1$ 的负例点，支持向量在超平面

$$H_2: w\cdot x+b=-1$$

上. 如图 7.3 所示，在 H_1 和 H_2 上的点就是支持向量.

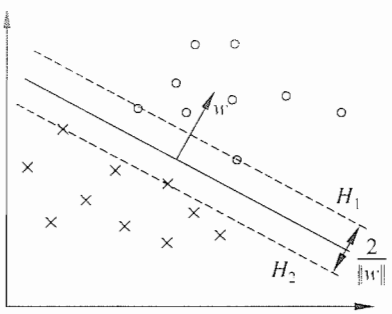

图 7.3 支持向量

注意到 H_1 和 H_2 平行，并且没有实例点落在它们中间. 在 H_1 与 H_2 之间形成一条长带，分离超平面与它们平行且位于它们中央. 长带的宽度，即 H_1 与 H_2 之间的距离称为间隔（margin）. 间隔依赖于分离超平面的法向量 w，等于 $\frac{2}{\|w\|}$. H_1 和 H_2 称为间隔边界.

在决定分离超平面时只有支持向量起作用，而其他实例点并不起作用. 如果移动支持向量将改变所求的解；但是如果在间隔边界以外移动其他实例点，甚至去掉这些点，则解是不会改变的. 由于支持向量在确定分离超平面中起着决定性作用，所以将这种分类模型称为支持向量机. 支持向量的个数一般很少，所以支持向量机由很少的“重要的”训练样本确定.

例 7.1 数据与例 2.1 相同. 已知一个如图 7.4 所示的训练数据集，其正例点是 $x_1=(3,3)^{\mathrm{T}}$， $x_2=(4,3)^{\mathrm{T}}$，负例点是 $x_3=(1,1)^{\mathrm{T}}$，试求最大间隔分离超平面.

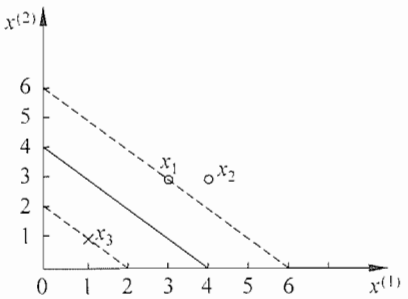

图 7.4 间隔最大分离超平面示例

解 按照算法 7.1，根据训练数据集构造约束最优化问题：

$$\begin{aligned}\min_{w,b}\quad & \frac{1}{2}(w_1^2+w_2^2)\\ \text{s.t.}\quad & 3w_1+3w_2+b\geqslant 1\\ & 4w_1+3w_2+b\geqslant 1\\ & -w_1-w_2-b\geqslant 1\end{aligned}$$

求得此最优化问题的解 $w_1=w_2=\frac{1}{2}$， $b=-2$. 于是最大间隔分离超平面为

$$\frac{1}{2}x^{(1)}+\frac{1}{2}x^{(2)}-2=0$$

其中， $x_1=(3,3)^{\mathrm{T}}$ 与 $x_3=(1,1)^{\mathrm{T}}$ 为支持向量. ■

7.1.4 学习的对偶算法

为了求解线性可分支持向量机的最优化问题 (7.13) ~ (7.14)，将它作为原始最优化问题，应用拉格朗日对偶性（参阅附录 C），通过求解对偶问题（dual problem）得到原始问题（primal problem）的最优解，这就是线性可分支持向量机的对偶算法（dual algorithm）. 这样做的优点，一是对偶问题往往更容易求解；二是自然引入核函数，进而推广到非线性分类问题.

首先构建拉格朗日函数（Lagrange function）. 为此，对每一个不等式约束 (7.14) 引进拉格朗日乘子（Lagrange multiplier）$\alpha_i\geqslant 0$， $i=1,2,\cdots,N$，定义拉格朗日函数：

$$L(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{N}\alpha_i y_i(w\cdot x_i+b)+\sum_{i=1}^{N}\alpha_i \tag{7.18}$$

其中，$\alpha=(\alpha_1,\alpha_2,\cdots,\alpha_N)^{\mathrm{T}}$ 为拉格朗日乘子向量.

根据拉格朗日对偶性，原始问题的对偶问题是极大极小问题：

$$\max_{\alpha}\min_{w,b}L(w,b,\alpha)$$

所以，为了得到对偶问题的解，需要先求 $L(w,b,\alpha)$ 对 w,b 的极小，再求对 α 的极大.

（1）求 $\min\limits_{w,b}L(w,b,\alpha)$

将拉格朗日函数 $L(w,b,\alpha)$ 分别对 w,b 求偏导数并令其等于 0.

$$\nabla_w L(w,b,\alpha)=w-\sum_{i=1}^{N}\alpha_i y_i x_i=0$$

$$\nabla_b L(w,b,\alpha)=\sum_{i=1}^{N}\alpha_i y_i=0$$

得

$$w=\sum_{i=1}^{N}\alpha_i y_i x_i \tag{7.19}$$

$$\sum_{i=1}^{N}\alpha_i y_i=0 \tag{7.20}$$

将式 (7.19) 代入拉格朗日函数 (7.18)，并利用式 (7.20)，即得

$$\begin{aligned}L(w,b,\alpha)&=\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)-\sum_{i=1}^{N}\alpha_i y_i\left(\left(\sum_{j=1}^{N}\alpha_j y_j x_j\right)\cdot x_i+b\right)+\sum_{i=1}^{N}\alpha_i\\&=-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)+\sum_{i=1}^{N}\alpha_i\end{aligned}$$

即

$$\min_{w,b}\ L(w,b,\alpha)=-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)+\sum_{i=1}^{N}\alpha_i$$

（2）求 $\min\limits_{w,b}\ L(w,b,\alpha)$ 对 α 的极大，即是对偶问题

$$\max_{\alpha}\ -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)+\sum_{i=1}^{N}\alpha_i \tag{7.21}$$

$$\text{s.t.}\quad \sum_{i=1}^{N}\alpha_i y_i=0$$

$$\alpha_i\geqslant 0,\quad i=1,2,\cdots,N$$

将式 (7.21) 的目标函数由求极大转换成求极小，就得到下面与之等价的对偶最优化问题：

$$\min_{\alpha} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{N}\alpha_i \tag{7.22}$$

$$\text{s.t.} \quad \sum_{i=1}^{N}\alpha_i y_i = 0 \tag{7.23}$$

$$\alpha_i \geqslant 0, \quad i = 1,2,\cdots,N \tag{7.24}$$

考虑原始最优化问题 (7.13) ～ (7.14) 和对偶最优化问题 (7.22) ～ (7.24)，原始问题满足定理 C.2 的条件，所以存在 w^*,α^*,β^*，使 w^* 是原始问题的解，α^*,β^* 是对偶问题的解．这意味着求解原始问题 (7.13) ～ (7.14) 可以转换为求解对偶问题(7.22) ～ (7.24)．

对线性可分训练数据集，假设对偶最优化问题 (7.22) ～ (7.24) 对 α 的解为 $\alpha^* = (\alpha_1^*,\alpha_2^*,\cdots,\alpha_N^*)^{\mathrm{T}}$，可以由 α^* 求得原始最优化问题 (7.13) ～ (7.14) 对 (w,b) 的解 w^*,b^*．有下面的定理．

定理 7.2 设 $\alpha^* = (\alpha_1^*,\alpha_2^*,\cdots,\alpha_l^*)^{\mathrm{T}}$ 是对偶最优化问题 (7.22) ～ (7.24) 的解，则存在下标 j，使得 $\alpha_j^* > 0$，并可按下式求得原始最优化问题 (7.13) ～ (7.14) 的解 w^*,b^*：

$$w^* = \sum_{i=1}^{N}\alpha_i^* y_i x_i \tag{7.25}$$

$$b^* = y_j - \sum_{i=1}^{N}\alpha_i^* y_i (x_i \cdot x_j) \tag{7.26}$$

证明 根据定理 C.3，KKT 条件成立，即得

$$\nabla_w L(w^*,b^*,\alpha^*) = w^* - \sum_{i=1}^{N}\alpha_i^* y_i x_i = 0 \tag{7.27}$$

$$\nabla_b L(w^*,b^*,\alpha^*) = -\sum_{i=1}^{N}\alpha_i^* y_i = 0$$

$$\alpha_i^*(y_i(w^* \cdot x_i + b^*) - 1) = 0, \quad i = 1,2,\cdots,N$$

$$y_i(w^* \cdot x_i + b^*) - 1 \geqslant 0, \quad i = 1,2,\cdots,N$$

$$\alpha_i^* \geqslant 0, \quad i = 1,2,\cdots,N$$

由此得

$$w^* = \sum_i \alpha_i^* y_i x_i$$

其中至少有一个 $\alpha_j^* > 0$（用反证法，假设 $\alpha^* = 0$，由式 (7.27) 可知 $w^* = 0$，而 $w^* = 0$ 不是原始最优化问题 (7.13) ～ (7.14) 的解，产生矛盾），对此 j 有

$$y_j(w^* \cdot x_j + b^*) - 1 = 0 \tag{7.28}$$

将式 (7.25) 代入式 (7.28) 并注意到 $y_j^2=1$，即得

$$b^*=y_j-\sum_{i=1}^{N}\alpha_i^* y_i(x_i\cdot x_j)$$

由此定理可知，分离超平面可以写成

$$\sum_{i=1}^{N}\alpha_i^* y_i(x\cdot x_i)+b^*=0 \tag{7.29}$$

分类决策函数可以写成

$$f(x)=\operatorname{sign}\left(\sum_{i=1}^{N}\alpha_i^* y_i(x\cdot x_i)+b^*\right) \tag{7.30}$$

这就是说，分类决策函数只依赖于输入 x 和训练样本输入的内积．式 (7.30) 称为线性可分支持向量机的对偶形式．

综上所述，对于给定的线性可分训练数据集，可以首先求对偶问题 (7.22) ～ (7.24) 的解 α^*；再利用式 (7.25) 和式 (7.26) 求得原始问题的解 w^*,b^*；从而得到分离超平面及分类决策函数．这种算法称为线性可分支持向量机的对偶学习算法，是线性可分支持向量机学习的基本算法．

算法 7.2（线性可分支持向量机学习算法）

输入：线性可分训练集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中 $x_i\in\mathcal{X}=\mathbf{R}^n$，$y_i\in\mathcal{Y}=\{-1,+1\}$，$i=1,2,\cdots,N$；

输出：分离超平面和分类决策函数．

（1）构造并求解约束最优化问题

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

$$\text{s.t.}\quad \sum_{i=1}^{N}\alpha_i y_i=0$$

$$\alpha_i\geqslant 0,\quad i=1,2,\cdots,N$$

求得最优解 $\alpha^*=(\alpha_1^*,\alpha_2^*,\cdots,\alpha_N^*)^{\mathrm{T}}$．

（2）计算

$$w^*=\sum_{i=1}^{N}\alpha_i^* y_i x_i$$

并选择 α^* 的一个正分量 $\alpha_j^*>0$，计算

$$b^*=y_j-\sum_{i=1}^{N}\alpha_i^* y_i(x_i\cdot x_j)$$

（3）求得分离超平面

$$w^* \cdot x + b^* = 0$$

分类决策函数：

$$f(x) = \text{sign}(w^* \cdot x + b^*)$$

在线性可分支持向量机中，由式 (7.25)、式 (7.26) 可知，w^* 和 b^* 只依赖于训练数据中对应于 $\alpha_i^* > 0$ 的样本点 (x_i, y_i)，而其他样本点对 w^* 和 b^* 没有影响．我们将训练数据中对应于 $\alpha_i^* > 0$ 的实例点 $x_i \in \mathbf{R}^n$ 称为支持向量．

定义 7.4（支持向量） 考虑原始最优化问题 (7.13) ～ (7.14) 及对偶最优化问题 (7.22) ～ (7.24)，将训练数据集中对应于 $\alpha_i^* > 0$ 的样本点 (x_i, y_i) 的实例 $x_i \in \mathbf{R}^n$ 称为支持向量．

根据这一定义，支持向量一定在间隔边界上．由 KKT 互补条件可知，

$$\alpha_i^* (y_i (w^* \cdot x_i + b^*) - 1) = 0, \quad i = 1, 2, \cdots, N$$

对应于 $\alpha_i^* > 0$ 的实例 x_i，有

$$y_i (w^* \cdot x_i + b^*) - 1 = 0$$

或

$$w^* \cdot x_i + b^* = \pm 1$$

即 x_i 一定在间隔边界上．这里的支持向量的定义与前面给出的支持向量的定义是一致的．

例 7.2 训练数据与例 7.1 相同．如图 7.4 所示，正例点是 $x_1 = (3,3)^{\mathrm{T}}$，$x_2 = (4,3)^{\mathrm{T}}$，负例点是 $x_3 = (1,1)^{\mathrm{T}}$，试用算法 7.2 求线性可分支持向量机．

解 根据所给数据，对偶问题是

$$\begin{aligned}
\min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{N}\alpha_i \\
& = \frac{1}{2}(18\alpha_1^2 + 25\alpha_2^2 + 2\alpha_3^2 + 42\alpha_1\alpha_2 - 12\alpha_1\alpha_3 - 14\alpha_2\alpha_3) - \alpha_1 - \alpha_2 - \alpha_3 \\
\text{s.t.} \quad & \alpha_1 + \alpha_2 - \alpha_3 = 0 \\
& \alpha_i \geqslant 0, \quad i = 1, 2, 3
\end{aligned}$$

解这一最优化问题．将 $\alpha_3 = \alpha_1 + \alpha_2$ 代入目标函数并记为

$$s(\alpha_1, \alpha_2) = 4\alpha_1^2 + \frac{13}{2}\alpha_2^2 + 10\alpha_1\alpha_2 - 2\alpha_1 - 2\alpha_2$$

对 α_1, α_2 求偏导数并令其为 0，易知 $s(\alpha_1, \alpha_2)$ 在点 $\left(\frac{3}{2}, -1\right)^{\mathrm{T}}$ 取极值，但该点不满足约束条件 $\alpha_2 \geqslant 0$，所以最小值应在边界上达到．

当 $\alpha_1=0$ 时，最小值 $s\left(0,\frac{2}{13}\right)=-\frac{2}{13}$；当 $\alpha_2=0$ 时，最小值 $s\left(\frac{1}{4},0\right)=-\frac{1}{4}$．于是 $s(\alpha_1,\alpha_2)$ 在 $\alpha_1=\frac{1}{4},\alpha_2=0$ 达到最小，此时 $\alpha_3=\alpha_1+\alpha_2=\frac{1}{4}$．

这样，$\alpha_1^*=\alpha_3^*=\frac{1}{4}$ 对应的实例点 x_1,x_3 是支持向量．根据式 (7.25) 和式 (7.26) 计算得

$$w_1^*=w_2^*=\frac{1}{2}$$

$$b^*=-2$$

分离超平面为

$$\frac{1}{2}x^{(1)}+\frac{1}{2}x^{(2)}-2=0$$

分类决策函数为

$$f(x)=\operatorname{sign}\left(\frac{1}{2}x^{(1)}+\frac{1}{2}x^{(2)}-2\right)$$ ■

对于线性可分问题，上述线性可分支持向量机的学习（硬间隔最大化）算法是完美的．但是，训练数据集线性可分是理想的情形．在现实问题中，训练数据集往往是线性不可分的，即在样本中出现噪声或特异点．此时，有更一般的学习算法．

7.2 线性支持向量机与软间隔最大化

7.2.1 线性支持向量机

线性可分问题的支持向量机学习方法，对线性不可分训练数据是不适用的，因为这时上述方法中的不等式约束并不能都成立．怎么才能将它扩展到线性不可分问题呢？这就需要修改硬间隔最大化，使其成为软间隔最大化．

假设给定一个特征空间上的训练数据集

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

其中，$x_i\in\mathcal{X}=\mathbf{R}^n$，$y_i\in\mathcal{Y}=\{+1,-1\}$，$i=1,2,\cdots,N$，$x_i$ 为第 i 个特征向量，y_i 为 x_i 的类标记．再假设训练数据集不是线性可分的．通常情况是，训练数据中有一些特异点（outlier），将这些特异点除去后，剩下大部分的样本点组成的集合是线性可分的．

线性不可分意味着某些样本点 (x_i,y_i) 不能满足函数间隔大于等于 1 的约束条件 (7.14)．为了解决这个问题，可以对每个样本点 (x_i,y_i) 引进一个松弛变量 $\xi_i\geqslant 0$，

使函数间隔加上松弛变量大于等于 1. 这样，约束条件变为

$$y_i(w\cdot x_i+b)\geqslant 1-\xi_i$$

同时，对每个松弛变量 ξ_i，支付一个代价 ξ_i. 目标函数由原来的 $\frac{1}{2}\|w\|^2$ 变成

$$\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\xi_i \tag{7.31}$$

这里，$C>0$ 称为惩罚参数，一般由应用问题决定，C 值大时对误分类的惩罚增大，C 值小时对误分类的惩罚减小. 最小化目标函数 (7.31) 包含两层含义：使 $\frac{1}{2}\|w\|^2$ 尽量小即间隔尽量大，同时使误分类点的个数尽量小，C 是调和二者的系数.

有了上面的思路，可以和训练数据集线性可分时一样来考虑训练数据集线性不可分时的线性支持向量机学习问题. 相应于硬间隔最大化，它称为软间隔最大化.

线性不可分的线性支持向量机的学习问题变成如下凸二次规划（convex quadratic programming）问题（原始问题）：

$$\min_{w,b,\xi}\quad \frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\xi_i \tag{7.32}$$

$$\text{s.t.}\quad y_i(w\cdot x_i+b)\geqslant 1-\xi_i,\quad i=1,2,\cdots,N \tag{7.33}$$

$$\xi_i\geqslant 0,\quad i=1,2,\cdots,N \tag{7.34}$$

原始问题 (7.32) ～ (7.34) 是一个凸二次规划问题，因而关于 (w,b,ξ) 的解是存在的. 可以证明 w 的解是唯一的，但 b 的解不唯一，b 的解存在于一个区间[1].

设问题 (7.32) ～ (7.34) 的解是 w^*，b^*，于是可以得到分离超平面 $w^*\cdot x+b^*=0$ 及分类决策函数 $f(x)=\text{sign}(w^*\cdot x+b^*)$. 称这样的模型为训练样本线性不可分时的线性支持向量机，简称为线性支持向量机. 显然，线性支持向量机包含线性可分支持向量机. 由于现实中训练数据集往往是线性不可分的，线性支持向量机具有更广的适用性.

下面给出线性支持向量机的定义.

定义 7.5（线性支持向量机） 对于给定的线性不可分的训练数据集，通过求解凸二次规划问题，即软间隔最大化问题 (7.32) ～ (7.34)，得到的分离超平面为

$$w^*\cdot x+b^*=0 \tag{7.35}$$

以及相应的分类决策函数

$$f(x)=\text{sign}(w^*\cdot x+b^*) \tag{7.36}$$

称为线性支持向量机.

7.2.2 学习的对偶算法

原始问题 (7.32) ～ (7.34) 的对偶问题是

$$\min_{\alpha} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{N}\alpha_i \tag{7.37}$$

$$\text{s.t.} \quad \sum_{i=1}^{N}\alpha_i y_i = 0 \tag{7.38}$$

$$0 \leqslant \alpha_i \leqslant C, \quad i = 1,2,\cdots,N \tag{7.39}$$

原始最优化问题 (7.32) ～ (7.34) 的拉格朗日函数是

$$L(w,b,\xi,\alpha,\mu) \equiv \frac{1}{2}\| w \|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i (y_i (w \cdot x_i + b) - 1 + \xi_i) - \sum_{i=1}^{N}\mu_i\xi_i \tag{7.40}$$

其中，$\alpha_i \geqslant 0, \mu_i \geqslant 0$.

对偶问题是拉格朗日函数的极大极小问题. 首先求 $L(w,b,\xi,\alpha,\mu)$ 对 w,b,ξ 的极小，由

$$\nabla_w L(w,b,\xi,\alpha,\mu) = w - \sum_{i=1}^{N}\alpha_i y_i x_i = 0$$

$$\nabla_b L(w,b,\xi,\alpha,\mu) = -\sum_{i=1}^{N}\alpha_i y_i = 0$$

$$\nabla_{\xi_i} L(w,b,\xi,\alpha,\mu) = C - \alpha_i - \mu_i = 0$$

得

$$w = \sum_{i=1}^{N}\alpha_i y_i x_i \tag{7.41}$$

$$\sum_{i=1}^{N}\alpha_i y_i = 0 \tag{7.42}$$

$$C - \alpha_i - \mu_i = 0 \tag{7.43}$$

将式 (7.41) ～ (7.43) 代入式 (7.40)，得

$$\min_{w,b,\xi} L(w,b,\xi,\alpha,\mu) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) + \sum_{i=1}^{N}\alpha_i$$

再对 $\min\limits_{w,b,\xi} L(w,b,\xi,\alpha,\mu)$ 求 α 的极大，即得对偶问题：

$$\max_{\alpha} \quad -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) + \sum_{i=1}^{N}\alpha_i \tag{7.44}$$

$$\text{s.t.} \quad \sum_{i=1}^{N}\alpha_i y_i = 0 \tag{7.45}$$

$$C-\alpha_i-\mu_i=0 \tag{7.46}$$

$$\alpha_i \geqslant 0 \tag{7.47}$$

$$\mu_i \geqslant 0, \quad i=1,2,\cdots,N \tag{7.48}$$

将对偶最优化问题 (7.44) ～ (7.48) 进行变换：利用等式约束 (7.46) 消去 μ_i，从而只留下变量 α_i，并将约束 (7.46) ～ (7.48) 写成

$$0 \leqslant \alpha_i \leqslant C \tag{7.49}$$

再将对目标函数求极大转换为求极小，于是得到对偶问题 (7.37) ～ (7.39).

可以通过求解对偶问题而得到原始问题的解，进而确定分离超平面和决策函数. 为此，就可以定理的形式叙述原始问题的最优解和对偶问题的最优解的关系.

定理 7.3 设 $\alpha^* = (\alpha_1^*, \alpha_2^*, \cdots, \alpha_N^*)^{\mathrm{T}}$ 是对偶问题 (7.37) ～ (7.39) 的一个解，若存在 α^* 的一个分量 α_j^*，$0 < \alpha_j^* < C$，则原始问题 (7.32) ～ (7.34) 的解 w^*, b^* 可按下式求得：

$$w^* = \sum_{i=1}^{N}\alpha_i^* y_i x_i \tag{7.50}$$

$$b^* = y_j - \sum_{i=1}^{N} y_i \alpha_i^* (x_i \cdot x_j) \tag{7.51}$$

证明 原始问题是凸二次规划问题，解满足 KKT 条件. 即得

$$\nabla_w L(w^*, b^*, \xi^*, \alpha^*, \mu^*) = w^* - \sum_{i=1}^{N}\alpha_i^* y_i x_i = 0 \tag{7.52}$$

$$\nabla_b L(w^*, b^*, \xi^*, \alpha^*, \mu^*) = -\sum_{i=1}^{N}\alpha_i^* y_i = 0$$

$$\nabla_\xi L(w^*, b^*, \xi^*, \alpha^*, \mu^*) = C - \alpha^* - \mu^* = 0$$

$$\alpha_i^* (y_i(w^* \cdot x_i + b^*) - 1 + \xi_i^*) = 0 \tag{7.53}$$

$$\mu_i^* \xi_i^* = 0 \tag{7.54}$$

$$y_i(w^* \cdot x_i + b^*) - 1 + \xi_i^* \geqslant 0$$

$$\xi_i^* \geqslant 0$$

$$\alpha_i^* \geqslant 0$$

$$\mu_i^* \geqslant 0, \quad i=1,2,\cdots,N$$

由式 (7.52) 易知式 (7.50) 成立. 再由式 (7.53) ～ (7.54) 可知，若存在 α_j^*，$0<\alpha_j^*<C$，则 $y_i(w^*\cdot x_i+b^*)-1=0$. 由此即得式 (7.51). ■

由此定理可知，分离超平面可以写成

$$\sum_{i=1}^{N}\alpha_i^* y_i(x\cdot x_i)+b^*=0 \tag{7.55}$$

分类决策函数可以写成

$$f(x)=\operatorname{sign}\left(\sum_{i=1}^{N}\alpha_i^* y_i(x\cdot x_i)+b^*\right) \tag{7.56}$$

式 (7.56) 为线性支持向量机的对偶形式.

综合前面的结果，有下面的算法.

算法 7.3（线性支持向量机学习算法）

输入：训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中，$x_i\in\mathcal{X}=\mathbf{R}^n$，$y_i\in\mathcal{Y}=\{-1,+1\}$，$i=1,2,\cdots,N$；

输出：分离超平面和分类决策函数.

（1）选择惩罚参数 $C>0$，构造并求解凸二次规划问题

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

$$\text{s.t.}\quad \sum_{i=1}^{N}\alpha_i y_i=0$$

$$0\leqslant\alpha_i\leqslant C,\quad i=1,2,\cdots,N$$

求得最优解 $\alpha^*=(\alpha_1^*,\alpha_2^*,\cdots,\alpha_N^*)^{\mathrm{T}}$.

（2）计算 $w^*=\sum_{i=1}^{N}\alpha_i^* y_i x_i$

选择 α^* 的一个分量 α_j^* 适合条件 $0<\alpha_j^*<C$，计算

$$b^*=y_j-\sum_{i=1}^{N}y_i\alpha_i^*(x_i\cdot x_j)$$

（3）求得分离超平面

$$w^*\cdot x+b^*=0$$

分类决策函数：

$$f(x)=\operatorname{sign}(w^*\cdot x+b^*)$$ ■

步骤 (2) 中，对任一适合条件 $0<\alpha_j^*<C$ 的 α_j^*，按式 (7.51) 都可求出 b^*，但是由于原始问题 (7.32) ～ (7.34) 对 b 的解并不唯一[1]，所以实际计算时可以取在所有

符合条件的样本点上的平均值.

7.2.3 支持向量

在线性不可分的情况下，将对偶问题 (7.37) ～ (7.39) 的解 $\alpha^* = (\alpha_1^*, \alpha_2^*, \cdots, \alpha_N^*)^{\mathrm{T}}$ 中对应于 $\alpha_i^* > 0$ 的样本点 (x_i, y_i) 的实例 x_i 称为支持向量（软间隔的支持向量）. 如图 7.5 所示，这时的支持向量要比线性可分时的情况复杂一些. 图中，分离超平面由实线表示，间隔边界由虚线表示，正例点由“○”表示，负例点由“×”表示. 图中还标出了实例 x_i 到间隔边界的距离 $\dfrac{\xi_i}{\|w\|}$.

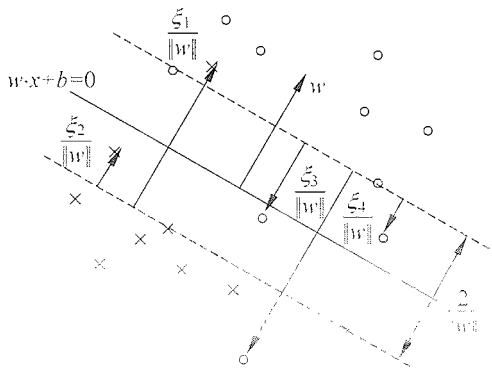

图 7.5 软间隔的支持向量

软间隔的支持向量 x_i 或者在间隔边界上，或者在间隔边界与分离超平面之间，或者在分离超平面误分一侧. 若 $\alpha_i^* < C$，则 $\xi_i = 0$，支持向量 x_i 恰好落在间隔边界上；若 $\alpha_i^* = C$，$0 < \xi_i < 1$，则分类正确，x_i 在间隔边界与分离超平面之间；若 $\alpha_i^* = C$，$\xi_i = 1$，则 x_i 在分离超平面上；若 $\alpha_i^* = C$，$\xi_i > 1$，则 x_i 位于分离超平面误分一侧.

7.2.4 合页损失函数

对于线性支持向量机学习来说，其模型为分离超平面 $w^* \cdot x + b^* = 0$ 及决策函数 $f(x) = \mathrm{sign}(w^* \cdot x + b^*)$，其学习策略为软间隔最大化，学习算法为凸二次规划.

线性支持向量机学习还有另外一种解释，就是最小化以下目标函数：

$$\sum_{i=1}^{N} [1 - y_i(w \cdot x_i + b)]_+ + \lambda \|w\|^2 \tag{7.57}$$

目标函数的第 1 项是经验损失或经验风险，函数

$$L(y(w\cdot x+b))=[1-y(w\cdot x+b)]_+ \tag{7.58}$$

称为合页损失函数（hinge loss function）．下标“＋”表示以下取正值的函数．

$$[z]_+=\begin{cases}z, & z>0\\ 0, & z\leqslant 0\end{cases} \tag{7.59}$$

这就是说，当样本点(x_i,y_i)被正确分类且函数间隔（确信度）$y_i(w\cdot x_i+b)$大于 1 时，损失是 0，否则损失是$1-y_i(w\cdot x_i+b)$，注意到在图 7.5 中的实例点x_4被正确分类，但损失不是 0．目标函数的第 2 项是系数为λ的w的L_2范数，是正则化项．

定理 7.4 线性支持向量机原始最优化问题：

$$\min_{w,b,\xi}\quad \frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\xi_i \tag{7.60}$$

$$\text{s.t.}\quad y_i(w\cdot x_i+b)\geqslant 1-\xi_i,\quad i=1,2,\cdots,N \tag{7.61}$$

$$\xi_i\geqslant 0,\quad i=1,2,\cdots,N \tag{7.62}$$

等价于最优化问题

$$\min_{w,b}\quad \sum_{i=1}^{N}[1-y_i(w\cdot x_i+b)]_+ +\lambda\|w\|^2 \tag{7.63}$$

证明 可将最优化问题 (7.63) 写成问题 (7.60) ～ (7.62)．令

$$1-y_i(w\cdot x_i+b)=\xi_i,\quad \xi_i\geqslant 0 \tag{7.64}$$

则$y_i(w\cdot x_i+b)\geqslant 1$．于是$w,b,\xi_i$满足约束条件 (7.61) ～ (7.62)．由式 (7.64) 有，$[1-y_i(w\cdot x_i+b)]_+=[\xi_i]_+=\xi_i$，所以最优化问题 (7.63) 可写成

$$\min_{w,b}\quad \sum_{i=1}^{N}\xi_i+\lambda\|w\|^2$$

若取$\lambda=\dfrac{1}{2C}$，则

$$\min_{w,b}\quad \frac{1}{C}\left(\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\xi_i\right)$$

与式 (7.60) 等价．

反之，也可将最优化问题 (7.60) ～ (7.62) 表示成问题 (7.63)． ■

合页损失函数的图形如图 7.6 所示，横轴是函数间隔$y(w\cdot x+b)$，纵轴是损失．由于函数形状像一个合页，故名合页损失函数．

图中还画出 0-1 损失函数，可以认为它是二类分类问题的真正的损失函数，而合页损失函数是 0-1 损失函数的上界．由于 0-1 损失函数不是连续可导的，直接

优化由其构成的目标函数比较困难，可以认为线性支持向量机是优化由 0-1 损失函数的上界（合页损失函数）构成的目标函数．这时的上界损失函数又称为代理损失函数（surrogate loss function）．

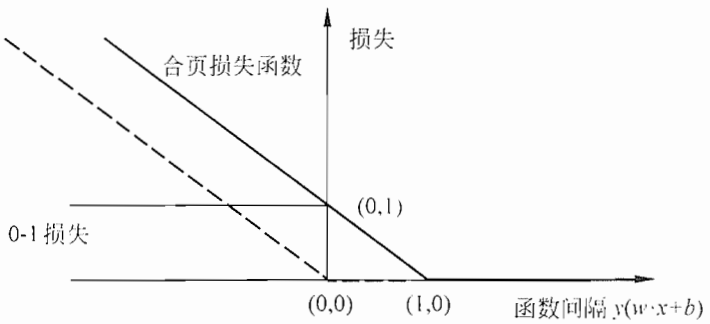

图 7.6 合页损失函数

图 7.6 中虚线显示的是感知机的损失函数 $[y_i(w\cdot x_i+b)]_+$．这时，当样本点 (x_i,y_i) 被正确分类时，损失是 0，否则损失是 $-y_i(w\cdot x_i+b)$．相比之下，合页损失函数不仅要分类正确，而且确信度足够高时损失才是 0．也就是说，合页损失函数对学习有更高的要求．

7.3 非线性支持向量机与核函数

对解线性分类问题，线性分类支持向量机是一种非常有效的方法．但是，有时分类问题是非线性的，这时可以使用非线性支持向量机．本节叙述非线性支持向量机，其主要特点是利用核技巧（kernel trick）．为此，先要介绍核技巧．核技巧不仅应用于支持向量机，而且应用于其他统计学习问题．

7.3.1 核技巧

1. 非线性分类问题

非线性分类问题是指通过利用非线性模型才能很好地进行分类的问题．先看一个例子：如 7.7 左图，是一个分类问题，图中“·”表示正实例点，“×”表示负实例点．由图可见，无法用直线（线性模型）将正负实例正确分开，但可以用一条椭圆曲线（非线性模型）将它们正确分开．

一般来说，对给定的一个训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中，实例 x_i 属于输入空间，$x_i\in\mathcal{X}=\mathbf{R}^n$，对应的标记有两类 $y_i\in\mathcal{Y}=\{-1,+1\}$，$i=1,2,\cdots,N$．如果能用 $\mathbf{R}^n$ 中的一个超曲面将正负例正确分开，则称这个问题为非线性可分问题．

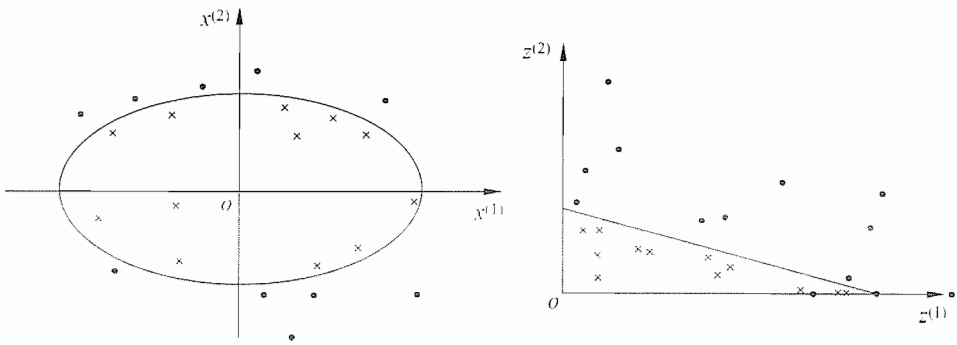

图 7.7 非线性分类问题与核技巧示例

非线性问题往往不好求解，所以希望能用解线性分类问题的方法解决这个问题. 所采取的方法是进行一个非线性变换，将非线性问题变换为线性问题，通过解变换后的线性问题的方法求解原来的非线性问题. 对图 7.7 所示的例子，通过变换，将左图中椭圆变换成右图中的直线，将非线性分类问题变换为线性分类问题.

设原空间为 $\mathcal{X} \subset \mathbf{R}^2$，$x=(x^{(1)},x^{(2)})^{\mathrm{T}} \in \mathcal{X}$，新空间为 $\mathcal{Z} \subset \mathbf{R}^2$，$z=(z^{(1)},z^{(2)})^{\mathrm{T}} \in \mathcal{Z}$，定义从原空间到新空间的变换（映射）：

$$z=\phi(x)=((x^{(1)})^2,(x^{(2)})^2)^{\mathrm{T}}$$

经过变换 $z=\phi(x)$，原空间 $\mathcal{X} \subset \mathbf{R}^2$ 变换为新空间 $\mathcal{Z} \subset \mathbf{R}^2$，原空间中的点相应地变换为新空间中的点，原空间中的椭圆

$$w_1(x^{(1)})^2+w_2(x^{(2)})^2+b=0$$

变换成为新空间中的直线

$$w_1z^{(1)}+w_2z^{(2)}+b=0$$

在变换后的新空间里，直线 $w_1z^{(1)}+w_2z^{(2)}+b=0$ 可以将变换后的正负实例点正确分开. 这样，原空间的非线性可分问题就变成了新空间的线性可分问题.

上面的例子说明，用线性分类方法求解非线性分类问题分为两步：首先使用一个变换将原空间的数据映射到新空间；然后在新空间里用线性分类学习方法从训练数据中学习分类模型. 核技巧就属于这样的方法.

核技巧应用到支持向量机，其基本想法就是通过一个非线性变换将输入空间（欧氏空间 $\mathbf{R}^n$ 或离散集合）对应于一个特征空间（希尔伯特空间 $\mathcal{H}$），使得在输入空间 $\mathbf{R}^n$ 中的超曲面模型对应于特征空间 $\mathcal{H}$ 中的超平面模型（支持向量机）. 这样，分类问题的学习任务通过在特征空间中求解线性支持向量机就可以完成.

2. 核函数的定义

定义 7.6（核函数） 设 $\mathcal{X}$ 是输入空间（欧氏空间 $\mathbf{R}^n$ 的子集或离散集合），又

设 $\mathcal{H}$ 为特征空间（希尔伯特空间），如果存在一个从 $\mathcal{X}$ 到 $\mathcal{H}$ 的映射

$$\phi(x):\mathcal{X}\to\mathcal{H} \tag{7.65}$$

使得对所有 $x,z\in\mathcal{X}$，函数 $K(x,z)$ 满足条件

$$K(x,z)=\phi(x)\cdot\phi(z) \tag{7.66}$$

则称 $K(x,z)$ 为核函数，$\phi(x)$ 为映射函数，式中 $\phi(x)\cdot\phi(z)$ 为 $\phi(x)$ 和 $\phi(z)$ 的内积.

核技巧的想法是，在学习与预测中只定义核函数 $K(x,z)$，而不显式地定义映射函数 ϕ. 通常，直接计算 $K(x,z)$ 比较容易，而通过 $\phi(x)$ 和 $\phi(z)$ 计算 $K(x,z)$ 并不容易. 注意，ϕ 是输入空间 $\mathbf{R}^n$ 到特征空间 $\mathcal{H}$ 的映射，特征空间 $\mathcal{H}$ 一般是高维的，甚至是无穷维的. 可以看到，对于给定的核 $K(x,z)$，特征空间 $\mathcal{H}$ 和映射函数 ϕ 的取法并不唯一，可以取不同的特征空间，即便是在同一特征空间里也可以取不同的映射.

下面举一个简单的例子来说明核函数和映射函数的关系.

例 7.3 假设输入空间是 $\mathbf{R}^2$，核函数是 $K(x,z)=(x\cdot z)^2$，试找出其相关的特征空间 $\mathcal{H}$ 和映射 $\phi(x):\mathbf{R}^2\to\mathcal{H}$.

解 取特征空间 $\mathcal{H}=\mathbf{R}^3$，记 $x=(x^{(1)},x^{(2)})^{\mathrm{T}}$，$z=(z^{(1)},z^{(2)})^{\mathrm{T}}$，由于

$$(x\cdot z)^2=(x^{(1)}z^{(1)}+x^{(2)}z^{(2)})^2=(x^{(1)}z^{(1)})^2+2x^{(1)}z^{(1)}x^{(2)}z^{(2)}+(x^{(2)}z^{(2)})^2$$

所以可以取映射

$$\phi(x)=((x^{(1)})^2,\sqrt{2}x^{(1)}x^{(2)},(x^{(2)})^2)^{\mathrm{T}}$$

容易验证 $\phi(x)\cdot\phi(z)=(x\cdot z)^2=K(x,z)$.

仍取 $\mathcal{H}=\mathbf{R}^3$ 以及

$$\phi(x)=\frac{1}{\sqrt{2}}((x^{(1)})^2-(x^{(2)})^2,2x^{(1)}x^{(2)},(x^{(1)})^2+(x^{(2)})^2)^{\mathrm{T}}$$

同样有 $\phi(x)\cdot\phi(z)=(x\cdot z)^2=K(x,z)$.

还可以取 $\mathcal{H}=\mathbf{R}^4$ 和

$$\phi(x)=((x^{(1)})^2,x^{(1)}x^{(2)},x^{(1)}x^{(2)},(x^{(2)})^2)^{\mathrm{T}}$$

■

3. 核技巧在支持向量机中的应用

我们注意到在线性支持向量机的对偶问题中，无论是目标函数还是决策函数（分离超平面）都只涉及输入实例与实例之间的内积. 在对偶问题的目标函数 (7.37) 中的内积 $x_i\cdot x_j$ 可以用核函数 $K(x_i,x_j)=\phi(x_i)\cdot\phi(x_j)$ 来代替. 此时对偶问题的目标函数成为

$$W(\alpha)=\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j)-\sum_{i=1}^{N}\alpha_i \tag{7.67}$$

同样，分类决策函数中的内积也可以用核函数代替，而分类决策函数式成为

$$f(x)=\operatorname{sign}\left(\sum_{i=1}^{N_s}a_i^* y_i\phi(x_i)\cdot\phi(x)+b^*\right)=\operatorname{sign}\left(\sum_{i=1}^{N_s}a_i^* y_i K(x_i,x)+b^*\right) \tag{7.68}$$

这等价于经过映射函数 ϕ 将原来的输入空间变换到一个新的特征空间，将输入空间中的内积 $x_i\cdot x_j$ 变换为特征空间中的内积 $\phi(x_i)\cdot\phi(x_j)$，在新的特征空间里从训练样本中学习线性支持向量机．当映射函数是非线性函数时，学习到的含有核函数的支持向量机是非线性分类模型．

也就是说，在核函数 $K(x,z)$ 给定的条件下，可以利用解线性分类问题的方法求解非线性分类问题的支持向量机．学习是隐式地在特征空间进行的，不需要显式地定义特征空间和映射函数．这样的技巧称为核技巧，它是巧妙地利用线性分类学习方法与核函数解决非线性问题的技术．在实际应用中，往往依赖领域知识直接选择核函数，核函数选择的有效性需要通过实验验证．

7.3.2 正定核

已知映射函数 ϕ，可以通过 $\phi(x)$ 和 $\phi(z)$ 的内积求得核函数 $K(x,z)$．不用构造映射 $\phi(x)$ 能否直接判断一个给定的函数 $K(x,z)$ 是不是核函数？或者说，函数 $K(x,z)$ 满足什么条件才能成为核函数？

本节叙述正定核的充要条件．通常所说的核函数就是正定核函数（positive definite kernel function）．为证明此定理先介绍有关的预备知识．

假设 $K(x,z)$ 是定义在 $\mathcal{X}\times\mathcal{X}$ 上的对称函数，并且对任意的 $x_1,x_2,\cdots,x_m\in\mathcal{X}$，$K(x,z)$ 关于 $x_1,x_2,\cdots,x_m$ 的 Gram 矩阵是半正定的．可以依据函数 $K(x,z)$，构成一个希尔伯特空间（Hilbert space），其步骤是：首先定义映射 ϕ 并构成向量空间 S；然后在 S 上定义内积构成内积空间；最后将 S 完备化构成希尔伯特空间．

1．定义映射，构成向量空间 S

先定义映射

$$\phi: x\to K(\cdot,x) \tag{7.69}$$

根据这一映射，对任意 $x_i\in\mathcal{X}$，$\alpha_i\in\mathbf{R}$，$i=1,2,\cdots,m$，定义线性组合

$$f(\cdot)=\sum_{i=1}^{m}\alpha_i K(\cdot,x_i) \tag{7.70}$$

考虑由线性组合为元素的集合 S．由于集合 S 对加法和数乘运算是封闭的，所以

S 构成一个向量空间.

2. 在 S 上定义内积，使其成为内积空间

在 S 上定义一个运算 $*$：对任意 $f,g\in S$，

$$f(\cdot)=\sum_{i=1}^{m}\alpha_i K(\cdot,x_i) \tag{7.71}$$

$$g(\cdot)=\sum_{i=1}^{l}\beta_j K(\cdot,z_j) \tag{7.72}$$

定义运算 $*$

$$f*g=\sum_{i=1}^{m}\sum_{j=1}^{l}\alpha_i\beta_j K(x_i,z_j) \tag{7.73}$$

证明运算 $*$ 是空间 S 的内积. 为此要证：

（1） $(cf)*g=c(f*g)$， $c\in\mathbf{R}$ (7.74)

（2） $(f+g)*h=f*h+g*h$， $h\in S$ (7.75)

（3） $f*g=g*f$ (7.76)

（4） $f*f\geqslant 0$， (7.77)

$$f*f=0\Leftrightarrow f=0 \tag{7.78}$$

其中，(1) ~ (3) 由式 (7.70) ~ 式 (7.72) 及 $K(x,z)$ 的对称性容易得到. 现证 (4) 之式 (7.77). 由式 (7.70) 及式 (7.73) 可得：

$$f*f=\sum_{i,j=1}^{m}\alpha_i\alpha_j K(x_i,x_j)$$

由 Gram 矩阵的半正定性知上式右端非负，即 $f*f\geqslant 0$.

再证 (4) 之式 (7.78). 充分性显然. 为证必要性，首先证明不等式：

$$|f*g|^2\leqslant(f*f)(g*g) \tag{7.79}$$

设 $f,g\in S$， $\lambda\in\mathbf{R}$，则 $f+\lambda g\in S$，于是，

$$(f+\lambda g)*(f+\lambda g)\geqslant 0$$

$$f*f+2\lambda(f*g)+\lambda^2(g*g)\geqslant 0$$

其左端是 λ 的二次三项式，非负，其判别式小于等于 0，即

$$(f*g)^2-(f*f)(g*g)\leqslant 0$$

于是式 (7.79) 得证. 现证若 $f*f=0$，则 $f=0$. 事实上，若

$$f(\cdot)=\sum_{i=1}^{m}\alpha_i K(\cdot,x_i)$$

则按运算$*$的定义式(7.73)，对任意的$x\in\mathcal{X}$，有

$$K(\cdot,x)*f=\sum_{i=1}^{m}\alpha_i K(x,x_i)=f(x)$$

于是，

$$|f(x)|^2=|K(\cdot,x)*f|^2 \tag{7.80}$$

由式(7.79)和式(7.77)有

$$|K(\cdot,x)*f|^2\leqslant(K(\cdot,x)*K(\cdot,x))(f*f)=K(x,x)(f*f)$$

由式(7.80)有

$$|f(x)|^2\leqslant K(x,x)(f*f)$$

此式表明，当$f*f=0$时，对任意的x都有$|f(x)|=0$.

至此，证明了$*$为向量空间$\mathcal{S}$的内积. 赋予内积的向量空间为内积空间，因此$\mathcal{S}$是一个内积空间. 既然$*$为$\mathcal{S}$的内积运算，那么仍然用$\cdot$表示，即若

$$f(\cdot)=\sum_{i=1}^{m}\alpha_i K(\cdot,x_i),\quad g(\cdot)=\sum_{i=1}^{l}\beta_j K(\cdot,z_j)$$

则

$$f\cdot g=\sum_{i=1}^{m}\sum_{j=1}^{l}\alpha_i\beta_j K(x_i,z_j) \tag{7.81}$$

3. 将内积空间$\mathcal{S}$完备化为希尔伯特空间

现在将内积空间$\mathcal{S}$完备化. 由式(7.81)定义的内积可以得到范数

$$\|f\|=\sqrt{f\cdot f} \tag{7.82}$$

因此，$\mathcal{S}$是一个赋范向量空间. 根据泛函分析理论，对于不完备的赋范向量空间$\mathcal{S}$，一定可以使之完备化，得到完备的赋范向量空间$\mathcal{H}$. 一个内积空间，当作为一个赋范向量空间是完备的时候，就是希尔伯特空间. 这样，就得到了希尔伯特空间$\mathcal{H}$.

这一希尔伯特空间$\mathcal{H}$称为再生核希尔伯特空间（reproducing kernel Hilbert space，RKHS). 这是由于核K具有再生性，即满足

$$K(\cdot,x)\cdot f=f(x) \tag{7.83}$$

及

$$K(\cdot,x)\cdot K(\cdot,z)=K(x,z) \tag{7.84}$$

称为再生核.

4. 正定核的充要条件

定理 7.5（正定核的充要条件） 设 $K:\mathcal{X}\times\mathcal{X}\to\mathbf{R}$ 是对称函数，则 $K(x,z)$ 为正定核函数的充要条件是对任意 $x_i\in\mathcal{X}$，$i=1,2,\cdots,m$，$K(x,z)$ 对应的 Gram 矩阵：

$$K=\left[K(x_i,x_j)\right]_{m\times m} \tag{7.85}$$

是半正定矩阵.

证明 必要性. 由于 $K(x,z)$ 是 $\mathcal{X}\times\mathcal{X}$ 上的正定核，所以存在从 $\mathcal{X}$ 到希尔伯特空间 $\mathcal{H}$ 的映射 ϕ，使得

$$K(x,z)=\phi(x)\cdot\phi(z)$$

于是，对任意 $x_1,x_2,\cdots,x_m$，构造 $K(x,z)$ 关于 $x_1,x_2,\cdots,x_m$ 的 Gram 矩阵

$$[K_{ij}]_{m\times m}=[K(x_i,x_j)]_{m\times m}$$

对任意 $c_1,c_2,\cdots,c_m\in\mathbf{R}$，有

$$\begin{aligned}\sum_{i,j=1}^{m}c_ic_jK(x_i,x_j)&=\sum_{i,j=1}^{m}c_ic_j(\phi(x_i)\cdot\phi(x_j))\\&=\left(\sum_i c_i\phi(x_i)\right)\cdot\left(\sum_j c_j\phi(x_j)\right)=\left\|\sum_i c_i\phi(x_i)\right\|^2\geqslant 0\end{aligned}$$

表明 $K(x,z)$ 关于 $x_1,x_2,\cdots,x_m$ 的 Gram 矩阵是半正定的.

充分性. 已知对称函数 $K(x,z)$ 对任意 $x_1,x_2,\cdots,x_m\in\mathcal{X}$，$K(x,z)$ 关于 $x_1,x_2,\cdots,x_m$ 的 Gram 矩阵是半正定的. 根据前面的结果，对给定的 $K(x,z)$，可以构造从 $\mathcal{X}$ 到某个希尔伯特空间 $\mathcal{H}$ 的映射：

$$\phi:x\to K(\cdot,x) \tag{7.86}$$

由式 (7.83) 可知，

$$K(\cdot,x)\cdot f=f(x)$$

并且

$$K(\cdot,x)\cdot K(\cdot,z)=K(x,z)$$

由式 (7.86) 即得

$$K(x,z)=\phi(x)\cdot\phi(z)$$

表明 $K(x,z)$ 是 $\mathcal{X}\times\mathcal{X}$ 上的核函数. ■

定理给出了正定核的充要条件，因此可以作为正定核，即核函数的另一定义.

定义 7.7（正定核的等价定义） 设 $\mathcal{X} \subset \mathbf{R}^n$，$K(x,z)$ 是定义在 $\mathcal{X} \times \mathcal{X}$ 上的对称函数，如果对任意 $x_i \in \mathcal{X}$，$i=1,2,\cdots,m$，$K(x,z)$ 对应的 Gram 矩阵

$$K=\left[K(x_i,x_j)\right]_{m\times m} \tag{7.87}$$

是半正定矩阵，则称 $K(x,z)$ 是正定核.

这一定义在构造核函数时很有用. 但对于一个具体函数 $K(x,z)$ 来说，检验它是否为正定核函数并不容易，因为要求对任意有限输入集 $\{x_1,x_2,\cdots,x_m\}$ 验证 K 对应的 Gram 矩阵是否为半正定的. 在实际问题中往往应用已有的核函数. 另外，由 Mercer 定理可以得到 Mercer 核（Mercer Kernel）[11]，正定核比 Mercer 核更具一般性. 下面介绍一些常用的核函数.

7.3.3 常用核函数

1. 多项式核函数（polynomial kernel function）

$$K(x,z)=(x\cdot z+1)^p \tag{7.88}$$

对应的支持向量机是一个 p 次多项式分类器. 在此情形下，分类决策函数成为

$$f(x)=\operatorname{sign}\left(\sum_{i=1}^{N_s} a_i^* y_i (x_i\cdot x+1)^p + b^*\right) \tag{7.89}$$

2. 高斯核函数（Gaussian kernel function）

$$K(x,z)=\exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right) \tag{7.90}$$

对应的支持向量机是高斯径向基函数（radial basis function）分类器. 在此情形下，分类决策函数成为

$$f(x)=\operatorname{sign}\left(\sum_{i=1}^{N_s} a_i^* y_i \exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right)+b^*\right) \tag{7.91}$$

3. 字符串核函数（string kernel function）

核函数不仅可以定义在欧氏空间上，还可以定义在离散数据的集合上. 比如，字符串核是定义在字符串集合上的核函数. 字符串核函数在文本分类、信息检索、生物信息学等方面都有应用.

考虑一个有限字符表 Σ．字符串 s 是从 Σ 中取出的有限个字符的序列，包括空字符串．字符串 s 的长度用 $|s|$ 表示，它的元素记作 $s(1)s(2)\cdots s(|s|)$．两个字符串 s 和 t 的连接记作 st．所有长度为 n 的字符串的集合记作 Σ^n，所有字符串的集合记作 $\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n$．

考虑字符串 s 的子串 u．给定一个指标序列 $i=(i_1,i_2,\cdots,i_{|u|})$，$1 \leqslant i_1 < i_2 < \cdots < i_{|u|} \leqslant |s|$，$s$ 的子串定义为 $u = s(i) = s(i_1)s(i_2)\cdots s(i_{|u|})$，其长度记作 $l(i) = i_{|u|} - i_1 + 1$．如果 i 是连续的，则 $l(i) = |u|$；否则，$l(i) > |u|$．

假设 S 是长度大于或等于 n 字符串的集合，s 是 S 的元素．现在建立字符串集合 S 到特征空间 $\mathcal{H}_n = R^{\Sigma^n}$ 的映射 $\phi_n(s)$．R^{Σ^n} 表示定义在 Σ^n 上的实数空间，其每一维对应一个字符串 $u \in \Sigma^n$，映射 $\phi_n(s)$ 将字符串 s 对应于空间 R^{Σ^n} 的一个向量，其在 u 维上的取值为

$$[\phi_n(s)]_u = \sum_{i:s(i)=u} \lambda^{l(i)} \tag{7.92}$$

这里，$0 < \lambda \leqslant 1$ 是一个衰减参数，$l(i)$ 表示字符串 i 的长度，求和在 s 中所有与 u 相同的子串上进行．

例如，假设 Σ 为英文字符集，n 为 3，S 为长度大于或等于 3 的字符串的集合．考虑将字符集 S 映射到特征空间 H_3，H_3 的一维对应于字符串 asd．这时，字符串“Nasdaq”与“lass das”在这一维上的值分别是 $[\phi_3(\text{Nasdaq})]_{asd} = \lambda^3$ 和 $[\phi_3(\text{lass□das})]_{asd} = 2\lambda^5$（□为空格）．在第 1 个字符串里，$asd$ 是连续的子串．在第 2 个字符串里，asd 是长度为 5 的不连续子串，共出现 2 次．

两个字符串 s 和 t 上的字符串核函数是基于映射 ϕ_n 的特征空间中的内积：

$$k_n(s,t) = \sum_{u \in \Sigma^n} [\phi_n(s)]_u [\phi_n(t)]_u = \sum_{u \in \Sigma^n} \sum_{(i,j):s(i)=t(j)=u} \lambda^{l(i)} \lambda^{l(j)} \tag{7.93}$$

字符串核函数 $k_n(s,t)$ 给出了字符串 s 和 t 中长度等于 n 的所有子串组成的特征向量的余弦相似度（cosine similarity）．直观上，两个字符串相同的子串越多，它们就越相似，字符串核函数的值就越大．字符串核函数可以由动态规划快速地计算．

7.3.4 非线性支持向量分类机

如上所述，利用核技巧，可以将线性分类的学习方法应用到非线性分类问题中去．将线性支持向量机扩展到非线性支持向量机，只需将线性支持向量机对偶形式中的内积换成核函数．

定义 7.8（非线性支持向量机） 从非线性分类训练集，通过核函数与软间隔最大化，或凸二次规划 (7.95) ～(7.97)，学习得到的分类决策函数

$$f(x)=\operatorname{sign}\left(\sum_{i=1}^{N}\alpha_i^* y_i K(x,x_i)+b^*\right) \tag{7.94}$$

称为非线性支持向量，$K(x,z)$ 是正定核函数.

下面叙述非线性支持向量机学习算法.

算法 7.4（非线性支持向量机学习算法）

输入：训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中 $x_i\in\mathcal{X}=\mathbf{R}^n$，$y_i\in\mathcal{Y}=\{-1,+1\}$，$i=1,2,\cdots,N$；

输出：分类决策函数.

（1）选取适当的核函数 $K(x,z)$ 和适当的参数 C，构造并求解最优化问题

$$\min_{\alpha}\quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j)-\sum_{i=1}^{N}\alpha_i \tag{7.95}$$

$$\text{s.t.}\quad \sum_{i=1}^{N}\alpha_i y_i=0 \tag{7.96}$$

$$0\leqslant\alpha_i\leqslant C,\quad i=1,2,\cdots,N \tag{7.97}$$

求得最优解 $\alpha^*=(\alpha_1^*,\alpha_2^*,\cdots,\alpha_N^*)^{\mathrm{T}}$.

（2）选择 α^* 的一个正分量 $0<\alpha_j^*<C$，计算

$$b^*=y_j-\sum_{i=1}^{N}\alpha_i^* y_i K(x_i\cdot x_j)$$

（3）构造决策函数：

$$f(x)=\operatorname{sign}\left(\sum_{i=1}^{N}\alpha_i^* y_i K(x\cdot x_i)+b^*\right)$$ ■

当 $K(x,z)$ 是正定核函数时，问题 (7.95) ～ (7.97) 是凸二次规划问题，解是存在的.

7.4 序列最小最优化算法

本节讨论支持向量机学习的实现问题. 我们知道，支持向量机的学习问题可以形式化为求解凸二次规划问题. 这样的凸二次规划问题具有全局最优解，并且有许多最优化算法可以用于这一问题的求解. 但是当训练样本容量很大时，这些算法往往变得非常低效，以致无法使用. 所以，如何高效地实现支持向量机学习就成为一个重要的问题. 目前人们已提出许多快速实现算法. 本节讲述其中的序列最小最优化（sequential minimal optimization，SMO）算法，这种算法 1998 年由 Platt 提出.

SMO 算法要解如下凸二次规划的对偶问题：

$$\min_{\alpha} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j)-\sum_{i=1}^{N}\alpha_i \tag{7.98}$$

$$\text{s.t.} \quad \sum_{i=1}^{N}\alpha_i y_i = 0 \tag{7.99}$$

$$0 \leqslant \alpha_i \leqslant C, \quad i=1,2,\cdots,N \tag{7.100}$$

在这个问题中，变量是拉格朗日乘子，一个变量 α_i 对应于一个样本点 (x_i,y_i)；变量的总数等于训练样本容量 N.

SMO 算法是一种启发式算法，其基本思路是：如果所有变量的解都满足此最优化问题的 KKT 条件（Karush-Kuhn-Tucker conditions），那么这个最优化问题的解就得到了. 因为 KKT 条件是该最优化问题的充分必要条件. 否则，选择两个变量，固定其他变量，针对这两个变量构建一个二次规划问题. 这个二次规划问题关于这两个变量的解应该更接近原始二次规划问题的解，因为这会使得原始二次规划问题的目标函数值变得更小. 重要的是，这时子问题可以通过解析方法求解，这样就可以大大提高整个算法的计算速度. 子问题有两个变量，一个是违反 KKT 条件最严重的那一个，另一个由约束条件自动确定. 如此，SMO 算法将原问题不断分解为子问题并对子问题求解，进而达到求解原问题的目的.

注意，子问题的两个变量中只有一个是自由变量. 假设 α_1，α_2 为两个变量，$\alpha_3,\alpha_4,\cdots,\alpha_N$ 固定，那么由等式约束 (7.99) 可知

$$\alpha_1 = -y_1\sum_{i=2}^{N}\alpha_i y_i$$

如果 α_2 确定，那么 α_1 也随之确定. 所以子问题中同时更新两个变量.

整个 SMO 算法包括两个部分：求解两个变量二次规划的解析方法和选择变量的启发式方法.

7.4.1 两个变量二次规划的求解方法

不失一般性，假设选择的两个变量是 α_1,α_2，其他变量 $\alpha_i(i=3,4,\cdots,N)$ 是固定的. 于是 SMO 的最优化问题 (7.98) ～ (7.100) 的子问题可以写成：

$$\begin{aligned}\min_{\alpha_1,\alpha_2} \quad & W(\alpha_1,\alpha_2)=\frac{1}{2}K_{11}\alpha_1^2+\frac{1}{2}K_{22}\alpha_2^2+y_1y_2K_{12}\alpha_1\alpha_2 \\ & -(\alpha_1+\alpha_2)+y_1\alpha_1\sum_{i=3}^{N}y_i\alpha_iK_{i1}+y_2\alpha_2\sum_{i=3}^{N}y_i\alpha_iK_{i2}\end{aligned} \tag{7.101}$$

$$\text{s.t.} \quad \alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{N} y_i \alpha_i = \varsigma \tag{7.102}$$

$$0 \leqslant \alpha_i \leqslant C, \quad i = 1,2 \tag{7.103}$$

其中，$K_{ij} = K(x_i, x_j), i, j = 1,2,\cdots,N$，$\varsigma$ 是常数，目标函数式 (7.101) 中省略了不含 α_1, α_2 的常数项.

为了求解两个变量的二次规划问题 (7.101) ～ (7.103)，首先分析约束条件，然后在此约束条件下求极小.

由于只有两个变量 (α_1, α_2)，约束可以用二维空间中的图形表示（如图 7.8 所示）.

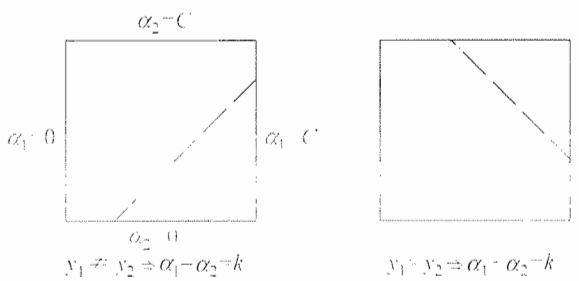

图 7.8 二变量优化问题图示

不等式约束 (7.103) 使得 (α_1, α_2) 在盒子 $[0,C]\times[0,C]$ 内，等式约束 (7.102) 使 (α_1, α_2) 在平行于盒子 $[0,C]\times[0,C]$ 的对角线的直线上. 因此要求的是目标函数在一条平行于对角线的线段上的最优值. 这使得两个变量的最优化问题成为实质上的单变量的最优化问题，不妨考虑为变量 α_2 的最优化问题.

假设问题 (7.101) ～ (7.103) 的初始可行解为 $\alpha_1^{\text{old}}, \alpha_2^{\text{old}}$，最优解为 $\alpha_1^{\text{new}}, \alpha_2^{\text{new}}$，并且假设在沿着约束方向未经剪辑时 α_2 的最优解为 $\alpha_2^{\text{new,unc}}$.

由于 α_2^{new} 需满足不等式约束 (7.103)，所以最优值 α_2^{new} 的取值范围必须满足条件

$$L \leqslant \alpha_2^{\text{new}} \leqslant H$$

其中，L 与 H 是 α_2^{new} 所在的对角线段端点的界. 如果 $y_1 \neq y_2$（如图 7.8 左图所示），则

$$L = \max(0, \alpha_2^{\text{old}} - \alpha_1^{\text{old}}), \quad H = \min(C, C + \alpha_2^{\text{old}} - \alpha_1^{\text{old}})$$

如果 $y_1 = y_2$（如图 7.8 右图所示），则

$$L = \max(0, \alpha_2^{\text{old}} + \alpha_1^{\text{old}} - C), \quad H = \min(C, \alpha_2^{\text{old}} + \alpha_1^{\text{old}})$$

下面，首先求沿着约束方向未经剪辑即未考虑不等式约束 (7.103) 时 α_2 的最优解 $\alpha_2^{\text{new,unc}}$；然后再求剪辑后 α_2 的解 α_2^{new}. 我们用定理来叙述这个结果. 为了叙述简单，记

$$g(x)=\sum_{i=1}^{N}\alpha_i y_i K(x_i,x)+b \tag{7.104}$$

令

$$E_i=g(x_i)-y_i=\left(\sum_{j=1}^{N}\alpha_j y_j K(x_j,x_i)+b\right)-y_i,\quad i=1,2 \tag{7.105}$$

当 $i=1,2$ 时，E_i 为函数 $g(x)$ 对输入 x_i 的预测值与真实输出 y_i 之差.

定理 7.6 最优化问题 (7.101) ～ (7.103) 沿着约束方向未经剪辑时的解是

$$\alpha_2^{\text{new,unc}}=\alpha_2^{\text{old}}+\frac{y_2(E_1-E_2)}{\eta} \tag{7.106}$$

其中，

$$\eta=K_{11}+K_{22}-2K_{12}=\left\|\Phi(x_1)-\Phi(x_2)\right\|^2 \tag{7.107}$$

$\Phi(x)$ 是输入空间到特征空间的映射，E_i，$i=1,2$，由式 (7.105) 给出.

经剪辑后 α_2 的解是

$$\alpha_2^{\text{new}}=\begin{cases}H, & \alpha_2^{\text{new,unc}}>H\\ \alpha_2^{\text{new,unc}}, & L\leqslant\alpha_2^{\text{new,unc}}\leqslant H\\ L, & \alpha_2^{\text{new,unc}}<L\end{cases} \tag{7.108}$$

由 α_2^{new} 求得 α_1^{new} 是

$$\alpha_1^{\text{new}}=\alpha_1^{\text{old}}+y_1y_2(\alpha_2^{\text{old}}-\alpha_2^{\text{new}}) \tag{7.109}$$

证明 引进记号

$$v_i=\sum_{j=3}^{N}\alpha_j y_j K(x_i,x_j)=g(x_i)-\sum_{j=1}^{2}\alpha_j y_j K(x_i,x_j)-b,\quad i=1,2$$

目标函数可写成

$$\begin{aligned}W(\alpha_1,\alpha_2)=&\frac{1}{2}K_{11}\alpha_1^2+\frac{1}{2}K_{22}\alpha_2^2+y_1y_2K_{12}\alpha_1\alpha_2\\&-(\alpha_1+\alpha_2)+y_1v_1\alpha_1+y_2v_2\alpha_2\end{aligned} \tag{7.110}$$

由 $\alpha_1y_1=\varsigma-\alpha_2y_2$ 及 $y_i^2=1$，可将 α_1 表示为

$$\alpha_1=(\varsigma-y_2\alpha_2)y_1$$

代入式 (7.110)，得到只是α_2的函数的目标函数：

$$W(\alpha_2)=\frac{1}{2}K_{11}(\varsigma-\alpha_2 y_2)^2+\frac{1}{2}K_{22}\alpha_2^2+y_2K_{12}(\varsigma-\alpha_2 y_2)\alpha_2$$
$$-(\varsigma-\alpha_2 y_2)y_1-\alpha_2+v_1(\varsigma-\alpha_2 y_2)+y_2v_2\alpha_2$$

对α_2求导数

$$\frac{\partial W}{\partial \alpha_2}=K_{11}\alpha_2+K_{22}\alpha_2-2K_{12}\alpha_2$$
$$-K_{11}\varsigma y_2+K_{12}\varsigma y_2+y_1y_2-1-v_1y_2+y_2v_2$$

令其为 0，得到

$$(K_{11}+K_{22}-2K_{12})\alpha_2=y_2(y_2-y_1+\varsigma K_{11}-\varsigma K_{12}+v_1-v_2)$$
$$=y_2\left[y_2-y_1+\varsigma K_{11}-\varsigma K_{12}+\left(g(x_1)-\sum_{j=1}^{2}y_j\alpha_jK_{1j}-b\right)\right.$$
$$\left.-\left(g(x_2)-\sum_{j=1}^{2}y_j\alpha_jK_{2j}-b\right)\right]$$

将$\varsigma=\alpha_1^{\text{old}}y_1+\alpha_2^{\text{old}}y_2$代入，得到

$$(K_{11}+K_{22}-2K_{12})\alpha_2^{\text{new,unc}}=y_2((K_{11}+K_{22}-2K_{12})\alpha_2^{\text{old}}y_2+y_2-y_1+g(x_1)-g(x_2))$$
$$=(K_{11}+K_{22}-2K_{12})\alpha_2^{\text{old}}+y_2(E_1-E_2)$$

将$\eta=K_{11}+K_{22}-2K_{12}$代入，于是得到

$$\alpha_2^{\text{new,unc}}=\alpha_2^{\text{old}}+\frac{y_2(E_1-E_2)}{\eta}$$

要使其满足不等式约束必须将其限制在区间$[L,H]$内，从而得到α_2^{new}的表达式 (7.108)．由等式约束 (7.102)，得到α_1^{new}的表达式 (7.109)．于是得到最优化问题 (7.101) ～ (7.103) 的解$(\alpha_1^{\text{new}},\alpha_2^{\text{new}})$．■

7.4.2 变量的选择方法

SMO 算法在每个子问题中选择两个变量优化，其中至少一个变量是违反 KKT 条件的．

1. 第 1 个变量的选择

SMO 称选择第 1 个变量的过程为外层循环．外层循环在训练样本中选取违反 KKT 条件最严重的样本点，并将其对应的变量作为第 1 个变量．具体地，检验训练样本点(x_i,y_i)是否满足 KKT 条件，即

$$\alpha_i=0\Leftrightarrow y_ig(x_i)\geqslant 1 \tag{7.111}$$

$$0<\alpha_i<C \Leftrightarrow y_i g(x_i)=1 \tag{7.112}$$

$$\alpha_i=C \Leftrightarrow y_i g(x_i)\leqslant 1 \tag{7.113}$$

其中，$g(x_i)=\sum_{j=1}^{N}\alpha_j y_j K(x_i,x_j)+b$.

该检验是在ε范围内进行的. 在检验过程中，外层循环首先遍历所有满足条件$0<\alpha_i<C$的样本点，即在间隔边界上的支持向量点，检验它们是否满足 KKT 条件. 如果这些样本点都满足 KKT 条件，那么遍历整个训练集，检验它们是否满足 KKT 条件.

2. 第 2 个变量的选择

SMO 称选择第 2 个变量的过程为内层循环. 假设在外层循环中已经找到第 1 个变量α_1，现在要在内层循环中找第 2 个变量α_2. 第 2 个变量选择的标准是希望能使α_2有足够大的变化.

由式 (7.106) 和式 (7.108) 可知，α_2^{new}是依赖于$|E_1-E_2|$的，为了加快计算速度，一种简单的做法是选择α_2，使其对应的$|E_1-E_2|$最大. 因为α_1已定，E_1也确定了. 如果E_1是正的，那么选择最小的E_i作为E_2；如果E_1是负的，那么选择最大的E_i作为E_2. 为了节省计算时间，将所有E_i值保存在一个列表中.

在特殊情况下，如果内层循环通过以上方法选择的α_2不能使目标函数有足够的下降，那么采用以下启发式规则继续选择α_2. 遍历在间隔边界上的支持向量点，依次将其对应的变量作为α_2试用，直到目标函数有足够的下降. 若找不到合适的α_2，那么遍历训练数据集；若仍找不到合适的α_2，则放弃第 1 个α_1，再通过外层循环寻求另外的α_1.

3. 计算阈值b和差值E_i

在每次完成两个变量的优化后，都要重新计算阈值b. 当$0<\alpha_1^{\text{new}}<C$时，由 KKT 条件 (7.112) 可知：

$$\sum_{i=1}^{N}\alpha_i y_i K_{i1}+b=y_1$$

于是，

$$b_1^{\text{new}}=y_1-\sum_{i=3}^{N}\alpha_i y_i K_{i1}-\alpha_1^{\text{new}} y_1 K_{11}-\alpha_2^{\text{new}} y_2 K_{21} \tag{7.114}$$

由E_1的定义式 (7.105) 有

$$E_1=\sum_{i=3}^{N}\alpha_i y_i K_{i1}+\alpha_1^{\text{old}} y_1 K_{11}+\alpha_2^{\text{old}} y_2 K_{21}+b^{\text{old}}-y_1$$

式 (7.114) 的前两项可写成：

$$y_1-\sum_{i=3}^{N}\alpha_i y_i K_{i1}=-E_1+\alpha_1^{\text{old}}y_1K_{11}+\alpha_2^{\text{old}}y_2K_{21}+b^{\text{old}}$$

代入式 (7.114)，可得

$$b_1^{\text{new}}=-E_1-y_1K_{11}(\alpha_1^{\text{new}}-\alpha_1^{\text{old}})-y_2K_{21}(\alpha_2^{\text{new}}-\alpha_2^{\text{old}})+b^{\text{old}} \tag{7.115}$$

同样，如果 $0<\alpha_2^{\text{new}}<C$，那么，

$$b_2^{\text{new}}=-E_2-y_1K_{12}(\alpha_1^{\text{new}}-\alpha_1^{\text{old}})-y_2K_{22}(\alpha_2^{\text{new}}-\alpha_2^{\text{old}})+b^{\text{old}} \tag{7.116}$$

如果 $\alpha_1^{\text{new}},\alpha_2^{\text{new}}$ 同时满足条件 $0<\alpha_i^{\text{new}}<C$，$i=1,2$，那么 $b_1^{\text{new}}=b_2^{\text{new}}$. 如果 α_1^{new}、α_2^{new} 是 0 或者 C，那么 b_1^{new} 和 b_2^{new} 以及它们之间的数都是符合 KKT 条件的阈值，这时选择它们的中点作为 b^{new}.

在每次完成两个变量的优化之后，还必须更新对应的 E_i 值，并将它们保存在列表中. E_i 值的更新要用到 b^{new} 值，以及所有支持向量对应的 α_j：

$$E_i^{\text{new}}=\sum_{S}y_j\alpha_jK(x_i,x_j)+b^{\text{new}}-y_i \tag{7.117}$$

其中，S 是所有支持向量 x_j 的集合.

7.4.3 SMO 算法

算法 7.5（SMO 算法）

输入：训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中，$x_i\in\mathcal{X}=\mathbf{R}^n$，$y_i\in\mathcal{Y}=\{-1,+1\}$，$i=1,2,\cdots,N$，精度 ε；

输出：近似解 $\hat{\alpha}$.

（1）取初值 $\alpha^{(0)}=0$，令 $k=0$；

（2）选取优化变量 $\alpha_1^{(k)},\alpha_2^{(k)}$，解析求解两个变量的最优化问题 (7.101) ～ (7.103)，求得最优解 $\alpha_1^{(k+1)},\alpha_2^{(k+1)}$，更新 α 为 $\alpha^{(k+1)}$；

（3）若在精度 ε 范围内满足停机条件

$$\sum_{i=1}^{N}\alpha_i y_i=0$$

$$0\leqslant\alpha_i\leqslant C,\quad i=1,2,\cdots,N$$

$$y_i\cdot g(x_i)=\begin{cases}\geqslant 1, & \{x_i\mid\alpha_i=0\}\\ =1, & \{x_i\mid 0<\alpha_i<C\}\\ \leqslant 1, & \{x_i\mid\alpha_i=C\}\end{cases}$$

其中，

$$g(x_i)=\sum_{j=1}^{N}\alpha_j y_j K(x_j,x_i)+b$$

则转 (4)；否则令 $k=k+1$，转 (2)；

（4）取 $\hat{\alpha}=\alpha^{(k+1)}$.

本 章 概 要

1．支持向量机最简单的情况是线性可分支持向量机，或硬间隔支持向量机．构建它的条件是训练数据线性可分．其学习策略是最大间隔法．可以表示为凸二次规划问题，其原始最优化问题为

$$\min_{w,b}\quad \frac{1}{2}\|w\|^2$$

$$\text{s.t.}\quad y_i(w\cdot x_i+b)-1\geqslant 0,\quad i=1,2,\cdots,N$$

求得最优化问题的解为 w^*，b^*，得到线性可分支持向量机，分离超平面是

$$w^*\cdot x+b^*=0$$

分类决策函数是

$$f(x)=\text{sign}(w^*\cdot x+b^*)$$

最大间隔法中，函数间隔与几何间隔是重要的概念．

线性可分支持向量机的最优解存在且唯一．位于间隔边界上的实例点为支持向量．最优分离超平面由支持向量完全决定．

二次规划问题的对偶问题是

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

$$\text{s.t.}\quad \sum_{i=1}^{N}\alpha_i y_i=0$$

$$\alpha_i\geqslant 0,\quad i=1,2,\cdots,N$$

通常，通过求解对偶问题学习线性可分支持向量机，即首先求解对偶问题的最优值 α^*，然后求最优值 w^* 和 b^*，得出分离超平面和分类决策函数．

2．现实中训练数据是线性可分的情形较少，训练数据往往是近似线性可分

的，这时使用线性支持向量机，或软间隔支持向量机．线性支持向量机是最基本的支持向量机．

对于噪声或例外，通过引入松弛变量ξ_i，使其“可分”，得到线性支持向量机学习的凸二次规划问题，其原始最优化问题是

$$\min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i$$

$$\text{s.t.} \quad y_i(w\cdot x_i + b) \geqslant 1-\xi_i, \quad i=1,2,\cdots,N$$

$$\xi_i \geqslant 0, \quad i=1,2,\cdots,N$$

求解原始最优化问题的解w^*，b^*，得到线性支持向量机，其分离超平面为

$$w^*\cdot x + b^* = 0$$

分类决策函数为

$$f(x) = \text{sign}(w^*\cdot x + b^*)$$

线性可分支持向量机的解w^*唯一但b^*不唯一．

对偶问题是

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

$$\text{s.t.} \quad \sum_{i=1}^{N}\alpha_i y_i = 0$$

$$0 \leqslant \alpha_i \leqslant C, \quad i=1,2,\cdots,N$$

线性支持向量机的对偶学习算法，首先求解对偶问题得到最优解α^*，然后求原始问题最优解w^*和b^*，得出分离超平面和分类决策函数．

对偶问题的解α^*中满足$\alpha_i^* > 0$的实例点x_i称为支持向量．支持向量可在间隔边界上，也可在间隔边界与分离超平面之间，或者在分离超平面误分一侧．最优分离超平面由支持向量完全决定．

线性支持向量机学习等价于最小化二阶范数正则化的合页函数

$$\sum_{i=1}^{N}\left[1-y_i(w\cdot x_i+b)\right]_+ + \lambda\|w\|^2$$

3．非线性支持向量机

对于输入空间中的非线性分类问题，可以通过非线性变换将它转化为某个高

维特征空间中的线性分类问题，在高维特征空间中学习线性支持向量机．由于在线性支持向量机学习的对偶问题里，目标函数和分类决策函数都只涉及实例与实例之间的内积，所以不需要显式地指定非线性变换，而是用核函数来替换当中的内积．核函数表示，通过一个非线性转换后的两个实例间的内积．具体地，$K(x,z)$是一个核函数，或正定核，意味着存在一个从输入空间$\mathcal{X}$到特征空间$\mathcal{H}$的映射$\phi(x):\mathcal{X}\to\mathcal{H}$，对任意$x,z\in\mathcal{X}$，有

$$K(x,z)=\phi(x)\cdot\phi(z)$$

对称函数$K(x,z)$为正定核的充要条件如下：对任意$x_i\in\mathcal{X}$，$i=1,2,\cdots,m$，任意正整数m，对称函数$K(x,z)$对应的Gram矩阵是半正定的．

所以，在线性支持向量机学习的对偶问题中，用核函数$K(x,z)$替代内积，求解得到的就是非线性支持向量机

$$f(x)=\text{sign}\left(\sum_{i=1}^{N}\alpha_i^*y_iK(x,x_i)+b^*\right)$$

4．SMO算法

SMO算法是支持向量机学习的一种快速算法，其特点是不断地将原二次规划问题分解为只有两个变量的二次规划子问题，并对子问题进行解析求解，直到所有变量满足KKT条件为止．这样通过启发式的方法得到原二次规划问题的最优解．因为子问题有解析解，所以每次计算子问题都很快，虽然计算子问题次数很多，但在总体上还是高效的．

继续阅读

线性支持向量机（软间隔）由Cortes与Vapnik提出[1]．同时，Boser, Guyon与Vapnik又引入核技巧，提出非线性支持向量机[2]．Drucker等人将其扩展到支持向量回归[3]．Vapnik Vladimir在他的统计学习理论[4]一书中对支持向量机的泛化能力进行了论述．

Platt提出了支持向量机的快速学习算法SMO[5]，Joachims实现的SVM Light，以及Chang与Lin实现的LIBSVM软件包被广泛使用．②

原始的支持向量机是二类分类模型，又被推广到多类分类支持向量机[6,7]，以及用于结构预测的结构支持向量机[8]．

关于支持向量机的文献很多．支持向量机的介绍可参照文献[9～12]．核方法

② SVM Light：http://svmlight.joachims.org/．LIBSVM：http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

被认为是比支持向量机更具一般性的机器学习方法. 核方法的介绍可参考文献[13～15].

习 题

1.1 比较感知机的对偶形式与线性可分支持向量机的对偶形式.

1.2 已知正例点 $x_1=(1,2)^{\mathrm{T}}$, $x_2=(2,3)^{\mathrm{T}}$, $x_3=(3,3)^{\mathrm{T}}$, 负例点 $x_4=(2,1)^{\mathrm{T}}$, $x_5=(3,2)^{\mathrm{T}}$, 试求最大间隔分离超平面和分类决策函数, 并在图上画出分离超平面、间隔边界及支持向量.

1.3 线性支持向量机还可以定义为以下形式:

$$\min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\xi_i^2$$

$$\text{s.t.} \quad y_i(w\cdot x_i+b)\geqslant 1-\xi_i, \quad i=1,2,\cdots,N$$

$$\xi_i\geqslant 0, \quad i=1,2,\cdots,N$$

试求其对偶形式.

1.4 证明内积的正整数幂函数:

$$K(x,z)=(x\cdot z)^p$$

是正定核函数, 这里 p 是正整数, $x,z\in\mathbf{R}^n$.

参 考 文 献

[1] Cortes C, Vapnik V. Support-vector networks. Machine Learning, 1995, 20

[2] Boser BE, Guyon IM, Vapnik VN. A training algorithm for optimal margin classifiers. In: Haussler D, ed. Proc of the 5th Annual ACM Workshop on COLT. Pittsburgh, PA, 1992, 144–152

[3] Drucker H, Burges CJC, Kaufman L, Smola A, Vapnik V. Support vector regression machines. In: Advances in Neural Information Processing Systems 9, NIPS 1996. MIT Press, 155–161

[4] Vapnik Vladimir N. The Nature of Statistical Learning Theory. Berlin: Springer-Verlag, 1995 (中译本: 张学工, 译. 统计学习理论的本质. 北京: 清华大学出版社, 2000)

[5] Platt JC. Fast training of support vector machines using sequential minimal optimization. Microsoft Research, http://research.microsoft.com/apps/pubs/?id=68391

[6] Weston JAE, Watkins C. Support vector machines for multi-class pattern recognition. In: Proceedings of the 7th European Symposium on Articial Neural Networks. 1999

[7] Crammer K, Singer Y. On the algorithmic implementation of multiclass kernel-based machines. Journal of Machine Learning Research, 2001, 2 (Dec): 265–292

[8] Tsochantaridis I, Joachims T, Hofmann T, Altun Y. Large margin methods for structured and interdependent output variables. JMLR, 2005, 6: 1453–1484

[9] Burges JC. A tutorial on support vector machines for pattern recognition. Bell Laboratories, Lucent Technologies. 1997

[10] Cristianini N, Shawe-Taylor J. An Introduction to Support Vector Machines and Othre Kerner-Based Learning Methods. Cambridge University Press，2000（中译本：李国正，等译. 支持向量机导论. 北京：电子工业出版社，2004）

[11] 邓乃扬，田英杰. 数据挖掘中的新方法——支持向量机. 北京：科学出版社，2004

[12] 邓乃扬，田英杰. 支持向量机——理论，算法与拓展. 北京：科学出版社，2009

[13] Scholkpf B, Smola AJ. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, 2002

[14] Herbrich R. Learning Kernel Classifiers, Theory and Algorithms. The MIT Press, 2002

[15] Hofmann T, Scholkopf B, Smola AJ. Kernel methods in machine learning. The Annals of Statistics, 2008, 36(3): 1171–1220